This is to certify that the

dissertation entitled

ON RECOGNIZING AND TRACKING 3D CURVED OBJECTS FROM 2D IMAGES

presented by

Jin-Long J. Chen

has been accepted towards fulfillment
of the requirements for

Ph.D. degree in Computer Science

Major professor

Date 24 Apr 96

ON RECOGNIZING AND TRACKING 3D CURVED OBJECTS FROM 2D IMAGES

By

Jin-Long J. Chen

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Computer Science Department

1996

ABSTRACT

ON RECOGNIZING AND TRACKING 3D CURVED OBJECTS FROM 2D IMAGES

By

Jin-Long J. Chen

This thesis addresses the problem of recognizing arbitrary curved 3D objects from single 2D intensity images. We propose a model-based solution within the alignment paradigm involving three major schemes – modeling, matching, and indexing.

The modeling scheme consists of constructing model aspects for predicting the object contour seen from any viewpoint. The indexing scheme generates from image features hypotheses giving candidate model aspects and poses. A hypothesis grouping and ordering method is used to order model hypotheses based on prior knowledge of pre-stored models and the visual evidence of the observed objects such that the most likely model hypotheses are tested first. The matching scheme aligns candidate model edgemaps to the observed object edgemap and the results of alignment are used to support/refute model hypotheses. Due to the unavailability of salient features in objects with sculptured surfaces, matching is carried out by the Newton's method with Levenberg-Marquardt minimization. A hierarchical verification strategy

is incorporated in the matching scheme to quickly eliminate false model hypotheses from further consideration. Once a correct model is localized, a recognition success is declared and the whole verification procedure is terminated. If all candidate model hypotheses are refuted, a recognition failure is reported. When combined into an integrated system, these three schemes make progress toward improving accuracy and efficiency by pruning false model hypotheses and minimizing unnecessary verification tests.

A prototype implementation has been tested in experiments conducted on a database containing 658 model aspects to evaluate the performance of recognition on 20 arbitrary curved objects, either non-occluded or partially occluded, seen from several viewpoints. Bench tests and simulations show that a large database of many kinds of objects including polyhedra and sculptured objects can be handled accurately and efficiently. We have also applied this model-based alignment paradigm to tracking a single moving object in a scene from an image sequence. Experimental results indicate the viability of this tracking method. From these results, we conclude that the proposed recognition-by-alignment paradigm is a viable approach to object recognition and tracking.

To my mentor Joseph C. Berston

ACKNOWLEDGMENTS

I would like to take this opportunity to thank my major advisor, Professor George C. Stockman, for his constant encouragement, continuous financial support, and excellent guidance throughout the course of this work. His efforts in making this learning experience worthwhile are deeply appreciated.

Special thanks are reserved to my dissertation committee members, Professor Anil K. Jain, Professor John Weng, and Professor Herbert Reynolds. The valuable suggestions and comments extended by both Professor Anil K. Jain and Professor John Weng brought this study to a successful completion. A particular debt of gratitude is owed to Professor Herbert Reynolds for providing financial support in the early stage of my study at Michigan State University and also for serving on my dissertation committee.

I also wish to thank my fellow Computer Science graduate students at the Pattern Recognition and Image Processing Laboratory past and present – Jian-Chang Mao, Sharathcha Pankanti, Qian Huang, Marie-Pierre Dubuisson, Tim Newman, Sally Howden, Chitra Dorai, Yu Zhong, Yuntao Cui, Shaoyun Chen, Dan Swets, Karissa Miller, and many others for their assistance and friendship and for making

my stay here a very enjoyable one. In addition, I am grateful to Jason Sperber for his help in data acquisition. Financial support for this research from the Texas Instruments is gratefully acknowledged.

Finally, as the important people in my life, I would like to express my deepest gratitude to my parents for their understanding, love, and moral support and in particular my mentor, Joseph C. Berston, for his everlasting concern and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

One of the central problems of computer vision is the automatic recovery of the structure and properties of three-dimensional (3D) objects in a scene from a single or multiple two-dimensional (2D) images. The structure and properties sought include the *identities*, *positions*, and *orientations* of objects. The **model-based recognition paradigm** has emerged as a promising approach to this problem in that stored models can be used to match against objects in the scene. This model-based recognition paradigm is supported by human eye-brain biological processes where the image of a scene is perceived by the eyes and the retinotopic patterns of objects in the scene are recalled from memory [118]. This object recognition paradigm used to model human biological processes [98, 118] or computer vision processes (*e.g.*, [28, 42, 92]) assumes some internal representation of visual input and some "computational" capacity to match it against representations of models known to the viewer. Consequently, model-based object recognition systems can be distinguished or classified according to their model representation schemes and their object matching techniques [12, 31].

This chapter is organized with the next section introducing the problem of recognizing 3D objects from 2D images. A few areas where opportunities of improvement over current recognition systems exist are also outlined. Section 1.2 describes the approach proposed in this thesis toward the object recognition problem. Since this thesis is about model-based object recognition by *alignment*, Section 1.3 surveys some of the most popular matching techniques related to alignment in use by the computer vision community. Section 1.4 outlines the major contributions of this thesis. The last section explains the organization and approach of this thesis.

1.1 Object Recognition by Alignment

Recognizing 3D objects from single 2D images has been an active research topic in computer vision since its inception. A practical solution to this problem has many applications such as inspection, automation, manipulation, navigation, security, *etc.*, and will greatly impact the field of intelligent robotics. Over the past twenty years, many researchers have studied the object recognition problem and have made progress in building experimental recognition systems. Most successful recognition systems use polyhedra as object models and rely on matching special object features to features extracted from the image. The most common strategy adopted in the computer vision community is known as *alignment* [60] or *hypothesize-and-verify* [4, 35, 42, 55, 76, 117]. This strategy divides the problem into two parts: first hypothesizing correspondences of features extracted from both the object model and the image, and then verifying the hypotheses. The verification process requires the position and

orientation (i.e., *pose*) of the hypothesized object be computed. Many researchers have developed techniques to make the search for correspondences more efficient by applying photometric and geometric constraint and have also studied the problem of pose estimation from feature correspondences. Despite these efforts, lack of efficiency and accuracy in alignment and the capacity of handling objects with arbitrary curved surfaces has caused current recognition systems to be less widely accepted in practice. We briefly discuss a few areas where new research is needed.

Simple Object Model: The most dominant reason for the popularity of polyhedra [40, 48, 60, 76] is that edge contours are relatively stable over small changes in viewpoint such that reliable geometric invariants/features can be extracted for hypothesis generation and verification purposes. Unfortunately, the world is not simple and polyhedral; it is complex and full of arbitrary curved objects whose image features are much more difficult to interpret. To be widely used in practice, recognition systems must accommodate sculptured surfaces and large model databases. Most researchers in computer vision have focused on object models with simple surfaces, namely polyhedra [40], quadric surfaces [108], superquadrics [5, 89], or super-ellipsoids [50, 88], or with slightly more complex ones such as algebraic surfaces [69, 64]. These surfaces are not expressive enough to represent sculptured objects. Even when sculptured objects are modeled by Boolean combinations of these primitives, they might not be smoothly sewn together.

Inefficient Verification Order: Hypothesis-generation procedures must compensate for inaccurate features extracted from the image (*e.g.*, due to noise or view

variation) by loosening constraints, thus increasing the number of incorrect hypotheses that are generated. Accordingly, optimizing the order in which the hypotheses are verified has become extremely important for the efficiency of recognition systems. Systems that do not optimize the ordering of hypotheses tend to verify more hypotheses than they should. Few systems make proper use of prior knowledge to guide the order of the verification so that the most likely hypotheses are verified first.

Inaccurate Pose Estimation: Correct verification of a hypothesis depends on accurate localization of the hypothesized object. Recognition systems that use correspondences of minimal sets of features to locate a hypothesized object often suffer from an inaccurate pose estimate. One remedy is to increase the tolerances for the verification decision; however, this often leads to more recognition errors. The other option is to refine the pose estimate using more available image features: unfortunately, approaches to such a pose refinement are often sensitive to missing features in the image (*e.g.*, due to partial occlusion), indicating that the refined pose estimate may not be an improvement unless the number of missing features is insignificant.

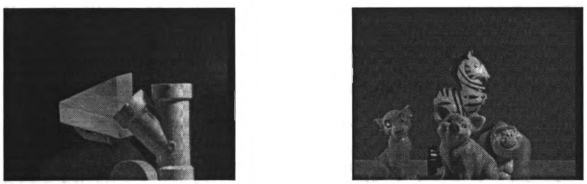
1.2 Problem Definition

In this thesis, we propose a model-based recognition system capable of handling a large database of 3D objects with sculptured ("free-form") surfaces. The proposed recognition system intends to solve the following problem:

> *Given a set of object models and an intensity image of a scene containing a jumble, specific objects in the scene are to be recognized based on the*

given single image.

This is the problem of recognizing objects with unknown 3D pose from a single 2D image. The object of interest may be partially occluded by other scene objects and the object models include both polyhedra and arbitrary curved objects. Examples of scenes are shown in Figure 1.1. The proposed recognition system fits within the theoretical framework for the recognition-by-alignment paradigm espoused by Ullman [111] and Lowe [75]. Alignment is a two-stage process. Given a model object and an image, the first stage is to hypothesize an aligning transformation that would bring the model object to a position and orientation in space that corresponds to the projected image. The second stage is to verify the hypothesis by applying the aligning transformation to the object model and then comparing the predicted appearance with the actual image. The degree of the match is used to refute/support the hypothesis, i.e., determine whether the image is in fact an instance of the model.



(a) (b)

Figure 1.1: Examples of scenes containing jumbles of objects.

The proposed approach toward object recognition strives to exploit prior knowledge to improve recognition performance and accuracy. Improvements are necessary

for object recognition systems to be practical for real applications. The proposed approach toward improving efficiency and accuracy of the recognition is based on (1) maximizing the use of prior knowledge to reduce unnecessary hypothesis verifications and (2) refining the pose estimate in a hill-climbing fashion to constrain the search. We represent prior knowledge using feature saliency in pre-stored models and the feature visual evidence in the images. For efficient recognition, likelihood measures based on prior knowledge are applied to select and order hypotheses for verification; they help remove relatively unlikely hypotheses from consideration and enable the proposed recognition system to verify the most likely hypotheses first. Robust pose refinement increases the accuracy of the pose estimate, and thus enhances the accuracy of the verification test as well.

The proposed object recognition system comprises three primary techniques: complex object modeling, probabilistic indexing, and robust matching.

Complex Object Modeling: The proposed modeling scheme builds/learns 3D models from sets of 2D images taken at controlled viewpoints. Our approach accounts for the changes of contour features over smooth object surfaces due to small changes in viewpoint. Curvature information about object surfaces is learnt from a set of images arising from the object, enabling the modeling scheme to generate the object's edge appearance seen from any viewpoint. Consequently, the proposed modeling scheme can handle both polyhedra and sculptured objects.

Probabilistic Indexing: The purpose of an indexing scheme is to use some image features to hypothesize candidate models and candidate poses. The proposed

indexing scheme generates hypotheses based on local geometric features because they are insensitive to partial occlusion. The proposed indexing scheme also increases the tolerances of indexing features to accommodate noise and view variation. The prior knowledge about the structure of pre-stored models and the visual evidence of image features is used to compute the relative likelihoods of the hypotheses. The likelihoods of these hypotheses are used to select hypotheses for verification by removing hypotheses with little supporting evidence while retaining those with strong supporting evidence. This means that incorrect hypotheses may be pruned because of their small likelihoods. The resulting hypotheses are ordered by their likelihoods so that the most likely hypotheses are tested first.

Robust Matching: The proposed matching technique refines the pose estimate in a hill-climbing fashion to constrain the search for correspondences between model and image features. We adopt a least-squares minimization approach—similar to the approach in [77]—to reliably estimate the object pose. This minimization technique uses heuristics to robustly estimate pose: it is shown to be relatively insensitive to outliers occurring due to partial occlusion.

The Recognition Task

The recognition problem addressed in this thesis is to identify and localize an arbitrary curved 3D object from a single 2D view of a 3D scene (i.e., 3D-to-2D recognition). Thus, an object has three translational and three rotational degrees of freedom that must be recovered from 2D sensory data. The input to the recognition system is an

intensity image and a set of aspect models. The output from the recognition system is an identity of the observed object and a transformation that best aligns the selected model with the observed image features.

The imaging process is assumed to be approximated by *weak perspective* projection, i.e., orthographic projection plus a scale factor. It is also assumed that objects are rigid. Under these assumptions, there are six unknown parameters for the image of an object under a rigid-body transformation: three for rotation, two for translation, and one for scaling [60].

Edge contour features are used for generating and verifying hypotheses. To allow occlusion, only portions of edge contours are used for generating and verifying hypotheses of candidate models and their pose in an image. Thus, it is assumed that objects are identifiable based on the shape of their edge contours. Since the proposed recognition system can handle objects that have smoothly changing surfaces, the edge contours are allowed to be slightly different over small changes in viewpoint.

A block diagram in Figure 1.2 depicts the proposed recognition paradigm and some major tasks addressed by this research. The modeling scheme consists of constructing model aspects. The indexing scheme generates from image features hypotheses giving candidate model aspects and poses which are then verified in the matching scheme to refute/support the hypotheses and also to estimate/refine the object pose of the correct model. The localization of a correct model implicitly indicates the recognition of the model. Once a correct model is localized, the whole verification procedure is terminated. If all candidate model hypotheses are refuted, a recognition failure is reported.

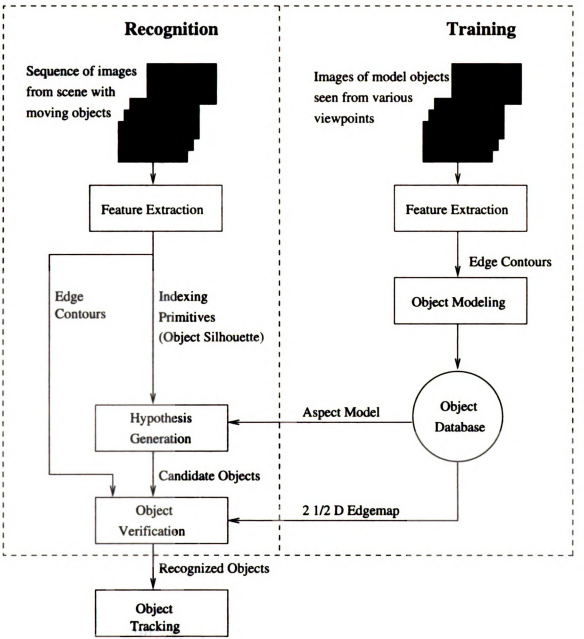


Figure 1.2: The computation paradigm of the proposed recognition system.

An example of the recognition of a partially occluded polyhedral object is given in Figure 1.3. Figure 1.3(a) shows a typical scene containing two objects: a polyhedra and a Y pipe. Figure 1.3(b) shows the edgemap of the image obtained from the feature extraction. Figure 1.3(c) shows the model (which is a $2\frac{1}{2}$D edgemap where the 3D information is embedded in each edge element) obtained from the object modeling scheme. Figure 1.3(d) depicts the evolution of convergence during the matching. Figure 1.3(e) shows the fit edgemap superimposed on the scene image, indicating that the polyhedron is recognized and located. Figure 1.3 does not show the indexing task, which is difficult to demonstrate by pictures.

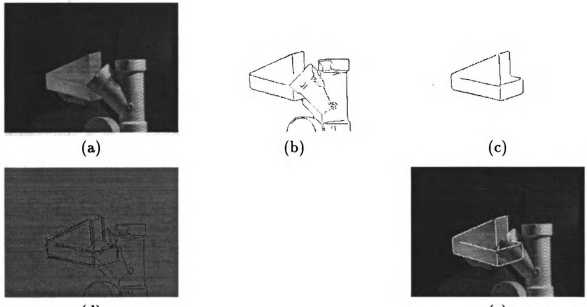


Figure 1.3: The major tasks of the proposed recognition paradigm. (a) A scene image containing an occluded polyhedra and a Y pipe. (b) The edge map of the scene obtained the feature extraction. (c) The polyhedra model obtained from object modeling is used for matching (note that the 3D information is embedded in each edge element). (d) The evolution of convergence during the matching. (e) The fitted edgemap shown superimposed on the scene image. (Note that the indexing task is not shown here.)

1.3 Matching Techniques Related to Alignment

All model-based recognition systems organize the recognition problem as a search for a match: specifically, how to associate features extracted from the sensory data with corresponding features extracted from the model [47]. Models can be constructed using salient geometric features such as arcs, lines, and corners. These features are local in nature and provide locational and rotational information to be used for computing the object pose and some contextual information for constraining the matching possibilities [104]. Thus, one way to structure the search problem is to consider the space of all possible matches of sensory features and model features. This space is defined as *correspondence space* [47]. An alternative is to directly search for a transformation in *parameter space* which causes each model feature to take on a position in the transformation coordinate system that coincides with the position of the corresponding sensory feature. Below, we briefly survey some of the existing matching techniques in the literature related to alignment for the purpose of object recognition. More reviews of related work can be found from surveys by Besl and Jain [12], Chin and Dyer [31], and Suetens *et al.* [106].

Interpretation Tree

The interpretation tree (IT) search is probably the most prominent recognition paradigm; see Grimson and Lozano-Perez [48]. The idea of interpretation tree search is to explore all possible correspondences of features between the sensory data and the model. A separate search is involved for each model, thus the search technique

itself does not involve indexing. Each node in the tree defines a matching pair of model feature and sensory data feature, and each path (from root to leaf node) of the tree defines an interpretation of the sensed features. An interpretation is considered to be *consistent* if there exists a rigid transformation which brings each model feature to coincide with the corresponding sensory feature. One very straightforward way of finding consistent interpretations is to simply do a systematic tree search of all branches (*e.g.*, backtracking or depth-first search). This brute-force tree search is exponential in the number of sensory features.

Several researchers have incorporated IT algorithms in alignment [17, 24, 40, 43, 49, 75]. The basic goal of these algorithms is to minimize the search by exploiting some prior knowledge to efficiently prune and order the search. If the first few chosen matches are accurate, the IT/Alignment algorithm can be very efficient. Grimson and Lozano-Perez [49] applied geometric (unary and binary) constraints to prune the search. Bolles and his collaborators [16, 17] developed the Local Feature Focus (LFF) method to avoid an exhaustive search. In this method, the most obvious feature matches (i.e., focus features) are used to guide the search for additional feature matches that will add the most information to the current interpretation, thus reducing the degrees of freedoms in the interpretation. Similar to the LFF method, Faugeras and Hebert [40] used the rigidity constraint to select subsequent matches in the interpretation tree. Flynn and Jain [43] used heuristic knowledge of the model database to prune and order the interpretation tree for efficient search. Lowe [75] introduced the use of perceptual grouping of features to order the search. Camps [24] took the approach of using probabilistic evidence to cut down the interpretation tree.

Interpretation tree search is a conservative approach since it considers all possibilities in the correspondence space; however, this method is robust at the expense of efficiency.

Geometric Hashing

While constrained search of the interpretation tree provides one method to reduce the search time, an alternative would be to use invariant features (defined on groups of features of the sensory data and the model) as indices for the construction and examination of tables containing the information needed to map the sensory feature to the model feature. This concept was introduced as "geometric hashing" by Lamdan and Wolfson [71, 72]. The method contains aspects of both indexing and matching and is based on the idea of representing an object by storing the object's transformation-invariant information in a hash table. At recognition time, similar invariants are derived from the sensory data, and are used to index into the hash table to retrieve candidate correspondences with the model. Each candidate correspondence constitutes a hypothesis whose validity is verified through pose information.

Pose Clustering

The *pose clustering* method [105, 103] uses aspects of both the correspondence space and the parameter space. The basic idea behind this method is to consider all possible pairs of corresponding sensed and model feature structures and then cluster on the parameter space of the pose transformation derived from each sensed-model structure

pair. The feature structure may be composed of (a) 3 surface normals, (b) 2 edges, or (c) 2 surface normals and 1 edge [104]. The task of matching is to put features in each sensory-model structure pair into correspondence based on local evidence and then derive the pose transformation in the process. Each pose transformation derived defines a point in pose space where a vote is cast on behalf of this sensory-model structure pairing. A cluster of matching pairs in this space is a good indication that many sensory data features are matched to corresponding model features. This method is fast but can miss good solutions due to the rough approximation of geometric constraints. It also wastes significant resources on regions of parameter space that could be easily determined not to contain a solution. Breuel [20] used adaptive subdivisions of parameter space to avoid such problems. He combined the idea of multiresolution matching, Hough transform, search-based recognition, and bounded error recognition into a simple, efficient algorithm, whose performance is better than that of alignment and pose clustering methods.

Alignment

Similar to the pose clustering method, the alignment method [60], or the hypothesize-and-verify method [4, 35, 55, 76], also uses aspects of both the correspondence space and the pose space. The basic idea is to find a minimal set of correspondences of sensory and model features sufficient to determine the pose and then to use the rigidity constraint to efficiently determine the other correspondences. For example, for 3D object localization, Huttenlocher and Ullman [60] use three pairs of data and model

points to deduce the six parameters of the transformation. Each such transformation constitutes a hypothesis about the pose of the object. The verification is done by aligning the model with the sensory features, that is, applying the hypothesized transformation to the model and using the transformed model to predict additional features that might be evident in the sensory data. This guides the search for additional supporting matches for pose refinement.

Pose Estimation via Model Fitting

A frequently required capability of an object recognition system is the localization of identified objects in the image. Several researchers have developed techniques to compute the object pose given the correspondences between model and image features (pose estimation). Pose estimation techniques can be used by the alignment search to generate an initial pose estimate. Typically, a few correspondences are used to hypothesize the initial pose estimate; other correspondences are then found by local search in the image, and the additional correspondences are used to refine the pose estimate. Pose estimation/refinement techniques normally involve minimization of some error function over the free model parameters. Lowe's [77] method minimizes the least-squared error using a technique based on Newton's method and Levenberg-Marquardt minimization to iteratively compute projection and model parameters that best fit a 3D model to 2D image contours. Kriegman and Ponce [69] used an elimination method to construct an implicit equation for image contours and then determined the model parameters by fitting the theoretical contour to the sensory

data. Ponce *et al.* [91] presented a global method for pose estimation by fitting an implicit algebraic surface to data points and a local method for pose refinement using the entire image contour. Haralick *et al.* [51] explored the use of robust weight functions in an iterative least-squared minimization for point-based pose estimation. Kass *et al.* [63] defined active contours, called *snakes*, that deform under "physical" constraints in an energy minimization procedure in which the equilibrium state determines the model parameters that yield the best match between the image and the model.

1.4 Contributions of the Thesis

This thesis proposes a complete model-based object recognition paradigm that can handle a large database of sculptured objects. A prototype implementation has been developed which is complete in the sense that given an input image, it is capable of detecting the identity and pose of an observed object. The system was designed to study the feasibility of an alignment approach to recognize 3D sculptured objects from 2D intensity images in a scene where the object of interest may be partially occluded. The system was also used to study applicability of the paradigm for tracking a single moving object in a scene from an image sequence. The system was tested on a database containing twenty 3D objects and eighty 2D objects with a total number of 658 model aspects. Since most object recognition systems reported in the literature can recognize only a small number of either unoccluded polyhedral objects from intensity images or curved objects from range images, the capacity of the new recog-

nition system to handle both polyhedra and sculptured objects in a more difficult scene is an important (overall) contribution of this thesis.

The first contribution of this thesis lies in the implementation of the curvature method proposed by Basri and Ullman [9] for object modeling. Our work both validates and extends the work of Basri and Ullman. We have explicitly included object internal edges in addition to object silhouette for object modeling. A 3D object is represented by a set of 2D images taken at controlled viewpoints. This model representation is descriptive in the sense that the edge contour of an object viewed from any given viewpoint can be predicted by using only a small number of viewer-center model aspects. Experimental results show the method is viable for modeling many kinds of objects including polyhedra and arbitrary curved objects.

The second contribution of this thesis lies in the design of an iterative matching technique and the use of two heuristics for matching. The pose estimation/refinement algorithm, which can handle objects with partial occlusion, does not assume the existence of salient features in the image, and thus, is directly applicable to smooth objects with sculptured surfaces. Due to the unavailability of salient features for correspondences, we develop an iterative matching technique for pose estimation/refinement. As with many other minimization techniques, this iterative matching technique requires good initial parameter estimates to avoid converging to a local minimum. From the experimental results, we demonstrate a high rate of convergence for a broad set of initial parameter estimates. The two heuristics are used to improve the pose accuracy and handle occlusion. The first heuristic is adopted to synthesize correspondences in order to refine the pose parameters in the hill-climbing procedure and to discard

certain edge segments in order to allow for some occlusion. Since the silhouettes of smooth objects have limited information, including internal edges in the fitting can help achieve more accurate pose. Thus, the second heuristic is used to maintain a balance in the importance of alignment of silhouette and internal edges such that the parameter fitting is not overly biased to one type of edge contour.

The third contribution of this thesis lies in the use of *part* invariant features for indexing and the use of hypothesis grouping and ordering for pruning false hypotheses. We show that the proposed part representation is robust under global transformation and in the presence of occlusion and spurious noise (i.e., local deformation). We have incorporated the pose consistency constraint to group into clusters the (parts) hypotheses within the same model. We have also investigated various hypothesis voting schemes which exploit prior knowledge of pre-stored models and the visual evidence of the observed objects to determine the order of hypotheses for verification. When combined into a complete system, these techniques make progress toward improving accuracy and efficiency by pruning false hypotheses and minimizing unnecessary verification tests.

The fourth contribution of this thesis lies in the use of a hierarchical verification strategy. Although the indexing scheme has effectively pruned false model hypotheses, the recognition system may still need to verify a few model hypotheses because the system loosens the indexing attribute tolerance to allow noise and view variation in the object silhouettes. This hierarchical verification strategy uses the fitting error as an indicator to reject false model hypotheses in the very early stage of the iterative matching, resulting in a significant reduction of the recognition time.

Another contribution of this thesis lies in the use of the recognition system for object tracking. We have applied the recognition system for tracking a single moving object in a scene from an image sequence. We have also studied how to track the moving object across aspects, and demonstrated its applicability by tracking a real moving car in an image sequence.

In Summary, the main contributions of this thesis are:

- ◇ A complete object recognition paradigm that can handle partially occluded objects with free-form surfaces in a database of 100 objects.
- ◇ A model representation that is compact and can handle both polyhedra and arbitrary curved objects.
- ◇ A robust iterative matching technique that does not require feature correspondences.
- ◇ An indexing scheme that can prune false model hypotheses effectively.
- ◇ A hierarchical verification strategy that speeds the recognition.
- ◇ The use of the recognition system for object tracking.

1.5 Organization of the Thesis

The remainder of this thesis is organized as follows. Chapter 2 presents our approach toward object modeling. It sets forth a very compact multi-view representation of 3D objects, and explains how this can be used to generate the object's edge appearance

from any viewpoint. The formal definition of model aspects is given. An example of a model aspect is given in Figure 1.4. We also describe the setup of our imaging configuration for the acquisition of aspect models. Experimental results show that the proposed modeling scheme can handle sculptured objects as well as polyhedra.

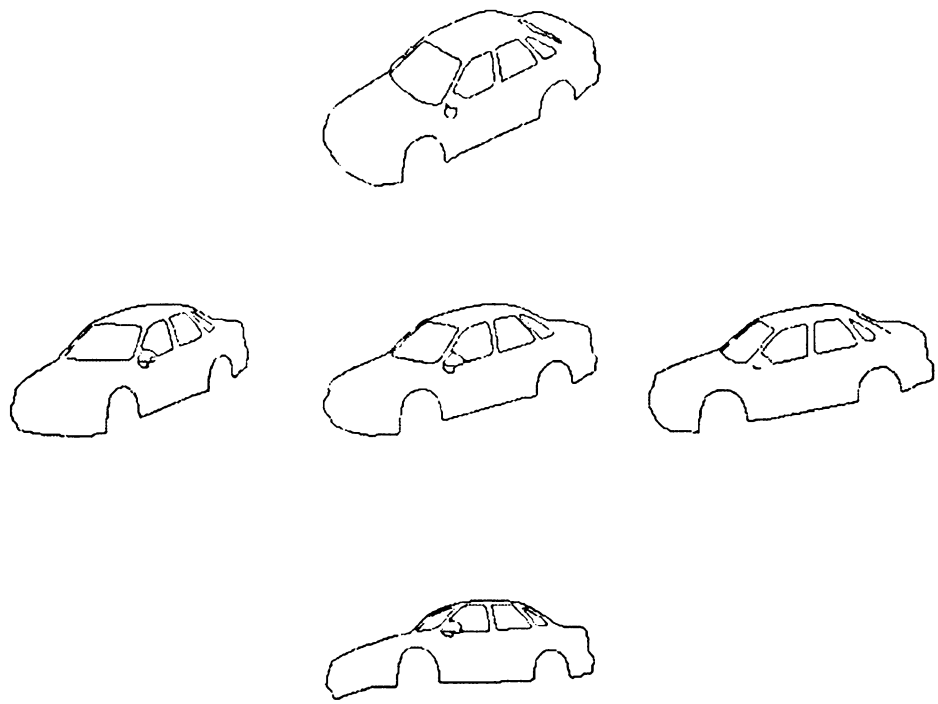


Figure 1.4: Prototype edgemaps for construction of a model aspect.

Matching 3D sculptured objects from a single 2D intensity image is treated in Chapter 3. It is well-known that inferring 3D object shapes from projected 2D silhouettes may not yield a unique solution. Symmetric objects may generate the same silhouette projections at various viewpoints. We propose an iterative matching technique which simultaneously fits object internal edges and object silhouette to the

edge contours of an image. Two heuristics are adopted in this iterative matching technique. One heuristic is used to synthesize correspondences in order to refine the pose in a hill-climbing fashion; the other heuristic is used to maintain a balance in the importance of silhouette and internal edges during the fitting. Results of fitting on a large set of real images are analyzed. We observe that fitting simultaneously internal edges and silhouette produces more accurate object pose than fitting silhouette alone. We conclude that our matching technique, although not perfect (mostly due to the lack of a perfect segmentation algorithm), can reliably localize a partially occluded 3D object from an image. An example of matching is depicted in Figure 1.5.

In Chapter 4, we present an indexing scheme that operates on object silhouettes. A robust *part* segmentation is introduced to decompose silhouettes into parts. The mathematical formulas for part invariant features are specified. We explain how the indexing scheme selects hypotheses using part invariant features. We present a hypothesis grouping scheme which clusters consistent part hypotheses that stem from the same model. We also present four different hypothesis voting schemes that order model hypotheses based on the pre-compiled part saliency and the visual evidence in the image. Experimental results indicate that the proposed indexing scheme generates a large number of model hypotheses, however, through hypothesis grouping and ordering, a significant number of verification tests for false model hypotheses can be avoided. An example is illustrated in Figure 1.6, where in our database of 658 model aspects, the scene of overlapped *elephant* and *bear* generates 5 model-aspect hypotheses where *elephant* is ranked ♯1 and *bear* ♯2 while the scene of occluded *deer* and *moose* produces 98 hypotheses with *moose* ranking ♯1 and *deer* ranking ♯2.

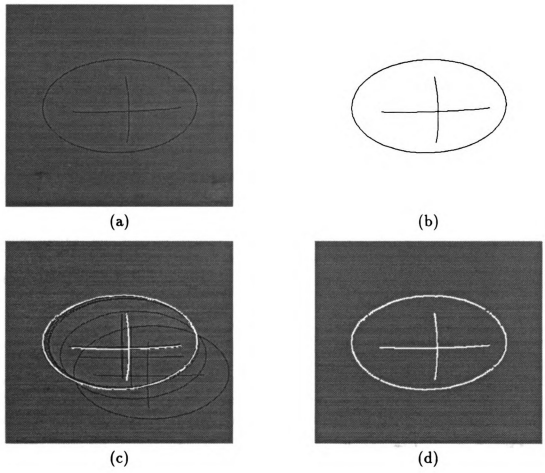


Figure 1.5: Steps in the matching algorithm. (a) Intensity image synthesized from a superquadric model. (b) Edgemap extracted from the image of (a). (c) The evolution of convergence in the matching method. (d) The fitted edgemap shown superimposed on the original image.

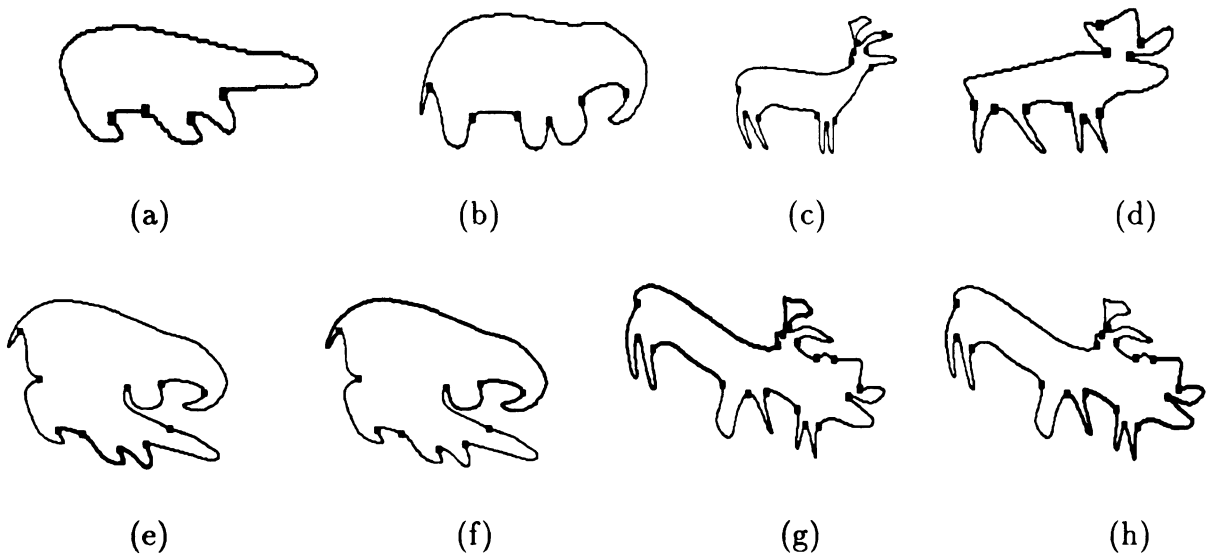


Figure 1.6: Hypothesis generation and pruning using part evidence. (a)–(d) Parts extracted from silhouette images of single bear, elephant, deer, and moose, respectively. (e)–(h) Thicker lines indicate parts used for generating hypotheses for bear, elephant, deer, and moose, respectively. (The images were obtained via anonymous FTP from the Institute for Robotics and Intelligent Systems, Department of Electrical Engineering, University of Southern California.)

In Chapter 5, we combine the modeling, matching, and indexing modules into a recognition system. We propose a hierarchical verification strategy to expedite the recognition. Experimental results show that false model hypotheses can be rejected at an early stage of verification and that the recognition system can efficiently recognize various kinds of objects. We also demonstrate the applicability of the complete recognition system for tracking a single moving object in an image sequence. An example of object tracking is illustrated in Figure 1.7.

Finally, Chapter 6 concludes the dissertation by evaluating the strengths and limitations of the proposed system, identifying some unresolved issues, and giving suggestions for future research.

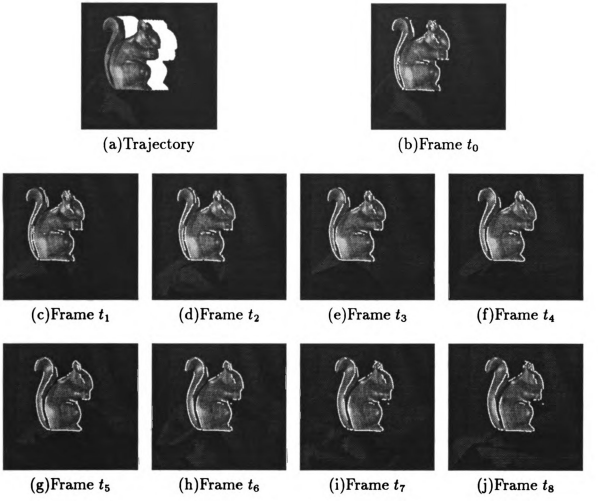


Figure 1.7: Object tracking of image sequences of a moving squirrel. (a) Image of frame t_0 added with thresholded images of frame t_1 - t_8. (b)-(j) are the results of model fitting with the white contours indicating the fitted edgemaps.

Chapter 2

Modeling 3D Objects from Multiple 2D Images

2.1 Introduction

This thesis is concerned with the recognition of rigid free-form 3D objects from a single intensity image. A model-based recognition system requires a representation that can model arbitrary solid objects to an appropriate level of detail and can provide abstract shape properties for matching purposes. A 3D object can be *implicitly* represented by a set of 2D projections taken from all possible viewpoints [47, 76, 93, 116], or *explicitly* represented by an object-centered geometric model [28, 29, 52, 121], or surface sweep and volume descriptions [13, 21, 79, 89]. Good descriptions of object representation may be found in the survey literature [12, 31].

The ultimate goal of vision is to recognize objects by seeing [78]. Many researchers in the computer vision community believe that a complete, object-centered, 3D model

must be built in order to do the recognition, because the model can be used to match against an instance of the model whenever it is viewed in a scene [48, 60, 75, 79]. Thus, most model-based vision systems for 3D object recognition require an object modeling module to provide appearances of an object observed from viewpoints such that the notable features and structural relations extracted from the sensed image and from the appearance of the model can be matched to determine the object's identity and pose.

An alternative for 3D object representation is a set of retinotopic 2D patterns. If all the possible 2D appearances of a 3D object are known, then recognition can be done by simply matching the image against all the 2D appearances stored in memory. However, this approach may give rise to the problem of requiring a very large number of views of each object, potentially every possible view of each object. As a consequence, this approach requires a large amount of memory, and different views of the same object are treated as distinct objects. One way to alleviate this memory burden is to describe objects by only representative views (i.e., aspects) in which features extracted from the images remain unchanged within the range covered by the view [39, 66]. For sculptured objects, the number of possible aspects may still be very large. The advantages of using retinotopic models are that the modeling scheme does not have to compute the vantage point and that the effect of perspective projection from a 3D model is compensated [92]. The disadvantages are that the representation is verbose and full of seams and that the indexing scheme must not only produce candidate objects but also particular views of that object. The success of this kind of modeling scheme for object recognition relies heavily on the completeness

of a full 3D representation of an object and the ability to index into a large set of model views. This completeness depends on the complexity of the object and the number of viewpoints used [31].

The approach adopted in this thesis is a hybrid of the aforementioned two approaches. We construct a 3D object model from a set of 2D intensity images taken from representative viewpoints tessellating the viewsphere. These viewpoints are defined as *aspects*. Each aspect defines a local coordinate system and is associated with five local images defined relative to this local coordinate system. The curvature method, devised by Basri and Ullman [9] for the construction of rigid objects bounded by smooth surfaces, uses these five local images to generate a $2\frac{1}{2}$D edgemap which can be used to produce the appearance of the observed object seen from any viewpoint within the aspect. The relative transformation between any two aspects is known and the 3D information of each aspect obtained from the curvature method is also available. This information can be fused together in order to build a complete 3D model in which the surface information is embedded in each aspect. However, building a complete model is not necessary for many problems; one exception is when used to track an object moving across aspects. Thus, in our approach, for recognition purposes, we neither have to store every possible view of an object, nor do we have to construct a complete geometric model.

This chapter is organized with the next section reviewing some existing techniques for modeling arbitrary curved objects. Section 2.3 presents the proposed object modeling scheme. Section 2.4 gives the implementation details. Section 2.5 shows results from the error analysis of the curvature method and from the alignment of models

with observed objects. The last section summarizes the proposed approach.

2.2 Related Background

We are interested in recognizing 3D sculptured objects from single 2D intensity images. Much past work on this problem has dealt with the easier model domains of polyhedra (*e.g.*, [75, 85]) and flat 3D objects (*e.g.*, [60, 71]), or has used other sensory information such as 3D range data (*e.g.*, [43, 52]) or multiple images (*e.g.*, [112]). In the following, we would like to review some of the existing techniques for modeling 3D sculptured objects with the capacity of generating 2D views for recognition.

The most straightforward approach is to model a 3D object by a set of 2D images taken from representative viewpoints. An object may be represented (1) by a set of 2D global feature vectors describing each possible stable view (*e.g.*, [93, 114, 116]), or (2) by a set of abstracted and precise geometric features (*e.g.*, [47, 76, 60, 103]). The advantages of the first method are that the representation is extremely compact as compared to other representations and that no correspondence between the sensory data and the model needs to be established for matching. The limitations are (1) that the representation is susceptible to occlusion and clutter because most features can only be extracted from isolated objects, and (2) that the object can not be accurately recreated or modeled from the features, nor can its pose be determined. The major problem of the representation in the second method lies in matching abstracted features. Features are local in nature, and each describes a portion of the object. Each feature also contains locational and rotational information to be

used to recover the object pose and some contextual information to constrain the matching possibilities. Localization can be used *implicitly* to solve the recognition problem. Since many local features are used to model an object, the search space for correspondences between image features and model features is exponential if some constraints are not enforced. Moreover, salient features, such as arcs, lines, and corners, may not be available for objects with sculptured surfaces to establish such correspondence. However, unlike the global feature vector representation, partial occlusion of objects can be handled.

Another promising approach is through building accurate geometric models of objects and then generating views by projecting the geometric model along the direction from the center of the viewsphere to a viewpoint. Some conventional CAD systems can represent man-made geometric models efficiently, but may not easily represent natural objects consisting of free-form surfaces. Several attempts have been made to develop methods for building geometric models from multi-view range images [30, 52, 67, 100] or a sequence of intensity images [70, 97, 113, 121]. For example, Higuchi *et al.* [52] presented a technique which uses a spherical representation, obtained by deforming a discrete mesh to fit the object surface, to merge multiple views into a complete object model. Seales and Faugeras [97] recovered a complete object surface from a sequence of images by fusing many views of the object extremal boundaries together using known object motion and object internal edges to place these views into a common coordinate frame.

Building geometric models from multi-view range images requires 3D data acquisition and view registration [15, 100] to form a consistent connected model. Some

optical range imaging sensors, such as a laser range finder, are ready to provide such range data from viewpoints. The relative positions of the sensor and the object can either be chosen adaptively based on the object geometry or the positions can be fixed. The approximate transformations between viewpoints provide useful information for merging data into a complete 3D data set. Range data obtained from passive range finders are normally contaminated with noises and holes (i.e., image locations with no depth values). Thus, constructing reliable free-form surfaces from range data is important. One popular approach, adopted by several researchers [30, 34, 52, 73, 80, 86], is to use deformable models to approximate a 3D surface by a mesh represented by some kind of tessellation of the viewsphere or by a B-spline type of mesh before deformation. During deformation, several force constraints, such as internal forces for maintaining the regularity and smoothness of the mesh and external forces for bringing the mesh to reach the object surface, are introduced to move the mesh toward the object surface until all the elements of the mesh "land" on the object surface. The registration estimates the relative transformation between viewpoints. Typically, the transformation is either assumed to be known, or it is solved by extracting features from each view and then establishing correspondences between them. Due to the unavailability of salient features from sculptured free-form surfaces, it is difficult to establish correspondences. This often results in iterative matching techniques [11, 29, 37] for bringing a set of 3D points from range data as close as possible to a set of primitives from a model. As is the case with any minimization technique, this type of technique requires a good initial estimate of parameters that is relatively close to the true transformation in order to avoid converging to a local

minimum.

Building geometric models from a sequence of intensity images requires that the observer's viewpoints be controlled in order to generate a dense sequence of images that allows incremental reconstruction of an unknown surface using the occluding contour [32, 46, 70, 97, 113]. Under continuous observer motion, the visible rim generated by the object's surface changes with the viewpoint, which affects the geometry of the occluding contour and also reveals the shape information for the parts of the surface over which the visible rim slides. A complete geometric model is then built by reconstructing many views of an object obtained from known motion and integrating local surface information over these views into a common coordinate system. The 3D model of the object is represented as an object-centered mesh that can be rotated and viewed from any direction.

2.3 Constructing Object Models

Object modeling is done for different purposes; models intended to support graphics or manufacturing might not support recognition without significant transformation of data. For recognition with pose detection, the object modeling module must efficiently provide as a function of viewpoint (pose parameters) the silhouette of the object and the internal edges created by surface marks, creases, or jumps. Edges caused by self-occluding limbs of the object have been ignored thus far in our implementation. Conceptually, they can be treated in the same manner as for limbs which create the silhouette; in practice, the current algorithm will consider them to be internal edges

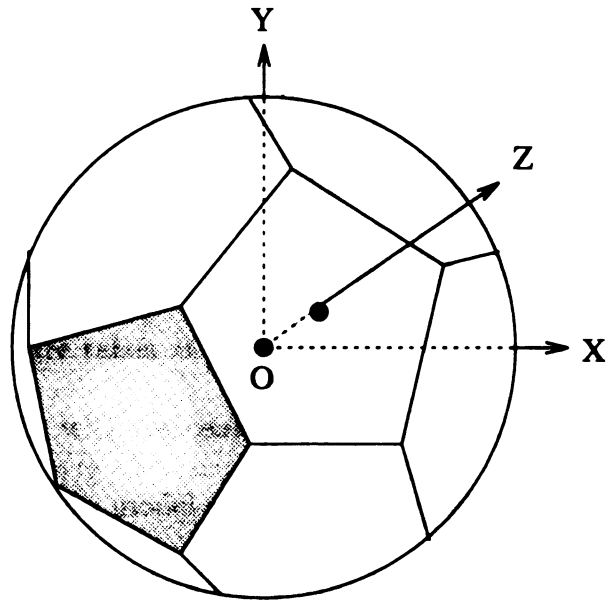


Figure 2.1: The viewing sphere can be tessellated into a regular polyhedra, each face defining an aspect.

and will ultimately discard them because of inconsistency in the stereo model. Some conventional CAD systems can readily provide such edgemaps, and, as is seen below, edgemaps can be extracted from superquadric models. In the rest of this section, we address the process of creating models from multiple intensity images of complex objects where use of a CAD system may be very difficult.

Our modeling of a 3D object using a set of 2D images proceeds as follows. Consider an object is placed at the center of a tessellated view sphere (see [56] for all the possible tessellations). Each face of the resulting polyhedron defines an *aspect* (see Figure 2.1). These *aspects* do not necessarily correspond to a topological similarity class, as has been common in the literature [39, 66]. Refinement of the tessellation could be used to increase homogeneity, but, since our matching is based on geometric criteria, we have not planned such refinements. For each model aspect, five images are used to create an edge-oriented $2\frac{1}{2}$D representation of the object. Each aspect or view defines

a **local coordinate system** with the Z-axis defined as the vector from the center of the sphere through the center of that polygon face, and the directions of the X-axis and Y-axis defined by the horizontal and vertical lines of the image, respectively. Of the five images, three are taken at the viewpoints on the sphere rotating around the X-axis, and three are taken at the viewpoints on the sphere rotating around the Y-axis (see Figure 2.2), with the central image common to both sets and taken along the Z-axis. An example of a model aspect is given in Figure 2.3. The model silhouette construction from these five images is specified in [9].

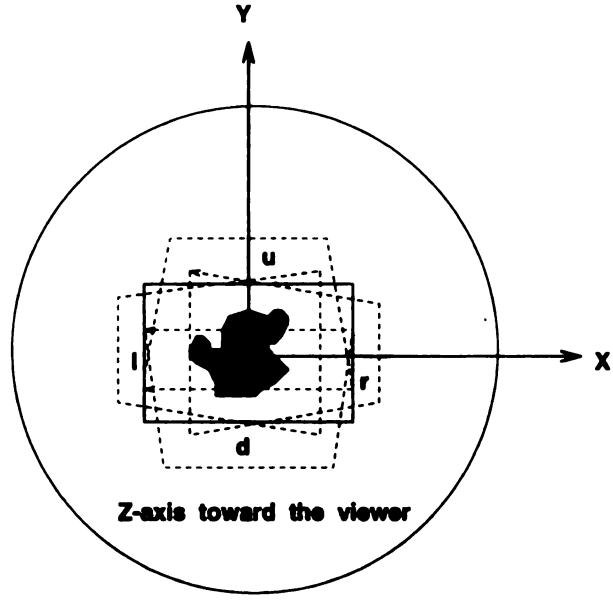


Figure 2.2: Model aspect made from 5 overlapping images.

It is important to emphasize that the object is modeled within an aspect in a standard rotation (roll angle) about the Z-axis which is normal to the optical axis of the central image used to create the aspect. Pan (rotation around the Y-axis in Figure 2.2) and tilt (rotation around the X-axis) can be quantized into intervals

to organize the model aspects around an entire viewsphere. Roll angle is unconstrained and thus must be approximately computed from image features to initialize the alignment optimization. Here, we assume a normalized zero-roll angle for the object. Determination of roll angle during recognition will be discussed in Chapter 4.

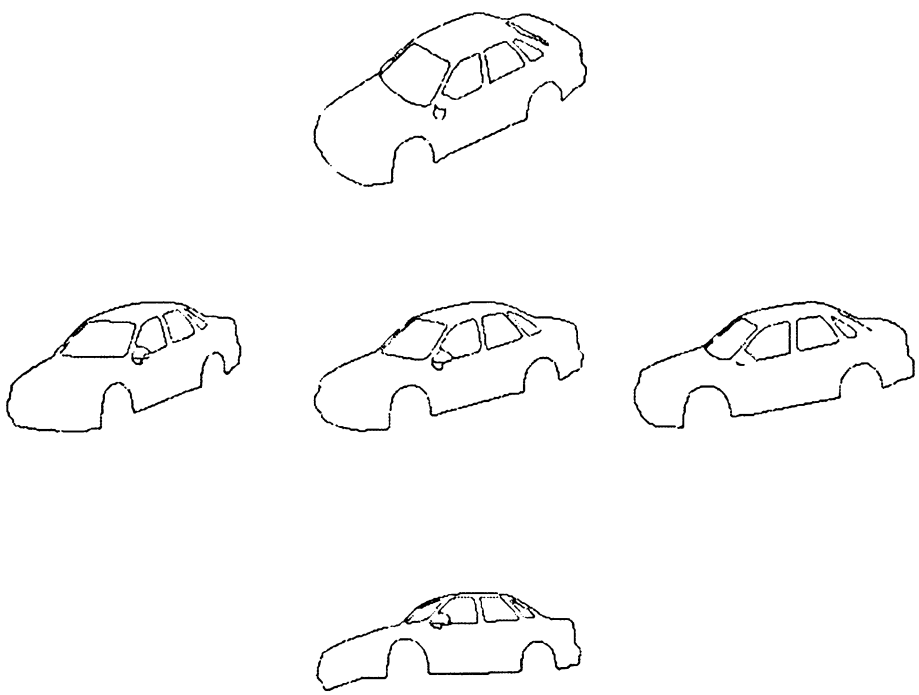


Figure 2.3: Example input for construction of a model aspect.

Model *aspects* do not need to cover the entire viewsphere in cases where certain views of the object are impossible, for example, from beneath a car. The informal requirement is that they cover the interesting views of an object and that they provide adequate approximation. Also, for some purpose, *e.g.*, recognition only or pose computation within a constrained environment, it is not necessary to register

the coordinate frame of each model aspect to a global coordinate frame. However, to continuously track an object with unconstrained pose, registration of each aspect frame to a global frame is needed. To be used for recognition, all viewpoints within an aspect should produce more or less the same features which will be used to index to that aspect. This means that tessellation of the viewsphere may be non-uniform in practice, due to variation in views. Most of our experiments have been done with a range of 20 degrees of rotation for one aspect. The appearance of an object at any given viewpoint can be generated by first indexing to the aspect of that viewpoint and then using that model aspect to predict the object appearance. Indexing to a model aspect will be treated in Chapter 4. We now turn to the mathematical formulas for constructing a model aspect from five neighboring training images.

2.3.1 Modeling the Object Silhouette

The curvature method of Basri and Ullman [9] is a technique for modeling and recovering an object's approximate edge appearance given a number of view-centered 2D projections. This modeling method is based on representing surface curvature of points along the silhouette of an image central to the aspect being modeled. This enables prediction of object silhouettes for viewpoints within that model aspect. The basic idea is depicted in Figure 2.4. For convenience, we sketch the curvature method adopted in this thesis.

Let X and Y be the main axes of the image plane, and let Z be the visual axis. Consider a smooth object rotating by a rotation **R** around the vertical axis Y. Let

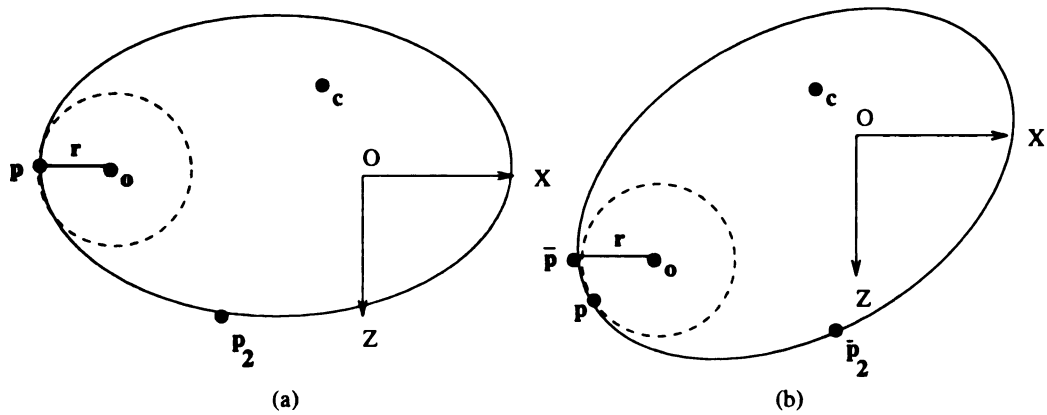


Figure 2.4: The curvature method. (a) A horizontal section of a smooth object. $\mathbf{p}$ is a point on the rim, $\mathbf{p_2}$ is an internal edge point, $\mathbf{r}$ is the curvature vector at $\mathbf{p}$, $\mathbf{o}$ is the center of the curvature circle, and $\mathbf{c}$ is the rotation center. The Z-axis is the visual axis, and the Y-axis points out of the paper. (b) The object is rotated about the Y-axis. $\bar{\mathbf{p}}$ is the new rim point approximated by Eq. (2.1) (based on [9]). $\bar{\mathbf{p}}_2$ is the corresponding point of $\mathbf{p_2}$ after rotation.

$\mathbf{p} = (x, y, z)$ be a rim point on the object. Let $\mathbf{r}$ be the curvature radius parallel to the X-axis. When the object is rotated by $\mathbf{R}$, point $\mathbf{p}$ may cease to be a rim point, and is replaced by a new rim point $\bar{\mathbf{p}} = (\bar{x}, \bar{y}, \bar{z})^t$ approximated by

$$\bar{\mathbf{p}} = \mathbf{R}(\mathbf{p} - \mathbf{c} - \mathbf{r}) + \mathbf{c} + \mathbf{r} \tag{2.1}$$

where $\mathbf{c} = (x_c, y_c, z_c)^t$ is the rotation center, and $\mathbf{r} = (r_x, r_y, 0)^t$ is the curvature vector at $\mathbf{p}$. The meaning of Eq. (2.1) is as follows. The point $\mathbf{o} = \mathbf{p} - \mathbf{r}$ is the center of the curvature circle. To predict the new rim point, we must first move $\mathbf{o}$ to the rotation center by $\mathbf{o} - \mathbf{c}$ and then apply $\mathbf{R}$ to $\mathbf{o} - \mathbf{c}$. Let $\bar{\mathbf{o}} = \mathbf{R}(\mathbf{p} - \mathbf{r} - \mathbf{c})$. The new rim point $\bar{\mathbf{p}}$ is obtained by translating $\bar{\mathbf{o}}$ with $\mathbf{c}$ and then $\mathbf{r}$ displacement. This method works well as long as the circle of curvature provides a good approximation to the section at $\mathbf{p}$. Under weak perspective projection, the z component of $\bar{\mathbf{p}}$ does not need to be

derived, thus, by modifying $\mathbf{r}$ to be $(r_x + x_c, r_y + y_c, 0)^t$ and $\mathbf{p}$ to be $(x, y, z - z_c)^t$, Eq. (2.1) can be changed into a simpler equation:

$$\bar{\mathbf{p}} = \mathbf{R}(\mathbf{p} - \mathbf{r}) + \mathbf{r}. \tag{2.2}$$

From Eq. (2.2), we can predict the position of $\bar{\mathbf{p}}$ for a rotation around an arbitrary axis in 3D space, provided that the radii of curvature r_x and r_y, and the relative depth $z - z_c$ at $\mathbf{p}$ are given. These parameters can be acquired from model construction [9] and are stored as parts of the object model.

2.3.2 Representing Internal Edges

Internal edges of an aspect are those visible at any viewpoint within the aspect and are caused by known discontinuities in albedo or surface normal or by artifacts such as unpredictable illumination or shadows. Unlike silhouette edges, internal edges are not usually occluded by the object surface due to a small rotation. Thus, corresponding points on internal edges before and after rotation exist in two images, and stereo matching can locate them in 3D. When an object is rotated by a 3×3 rotation matrix $\mathbf{R}$, an internal edge point $\mathbf{p} = (x, y, z)^t$ is transformed to a new position $\bar{\mathbf{p}} = (\bar{x}, \bar{y}, \bar{z})^t$ by

$$\bar{\mathbf{p}} = \mathbf{R}(\mathbf{p} - \mathbf{c}) + \mathbf{c}. \tag{2.3}$$

Note that Eq. (2.3) is a special case of Eq. (2.1) with the curvature vector $\mathbf{r} = \mathbf{0}$. Thus, Eq. (2.3) can also be modified to get the form of Eq. (2.2) by letting $\mathbf{r} = (x_c, y_c, 0)^t$

and $\mathbf{p} = (x, y, z - z_c)^t$.

2.3.3 Model Construction

We now show how to compute $\mathbf{r}$ using a 3-image stereo computation. Let A, B, and C denote three images taken from three different viewpoints in the aspect along a circle of the viewing sphere perpendicular to the Y-axis (see Figure 2.5). Let ζ be the rotation angle rotating a camera from viewpoint A to viewpoint B, and ν the rotation angle rotating a camera from viewpoint A to viewpoint C. Equivalently, the object may be rotated instead of the camera, which we often do. Let $\mathbf{p_1} = (x_1, y, z_1)^t$, $\mathbf{p_2} = (x_2, y, z_2)^t$ and $\mathbf{p_3} = (x_3, y, z_3)^t$ be three corresponding points in images A, B, and C, respectively. From Eq. (2.3), we have

$$x_2 = (x_1 - x_c)\cos\zeta + (z_1 - z_c)\sin\zeta + x_c, \tag{2.4}$$

$$x_3 = (x_1 - x_c)\cos\nu + (z_1 - z_c)\sin\nu + x_c. \tag{2.5}$$

Solving these two equations for the two unknown parameters x_c and $z_1 - z_c$ yields

$$z_1 - z_c = \frac{x_1(\cos\zeta - \cos\nu) - x_2(1 - \cos\nu) + x_3(1 - \cos\zeta)}{(1 - \cos\zeta)\sin\nu - \sin\zeta(1 - \cos\nu)}, \tag{2.6}$$

and

$$x_c = \frac{x_1\sin(\zeta - \nu) + x_2\sin\nu - x_3\sin\zeta}{(1 - \cos\zeta)\sin\nu - \sin\zeta(1 - \cos\nu)}. \tag{2.7}$$

With $-\frac{\pi}{2} < \zeta, \nu < \frac{\pi}{2}$ and $\zeta \neq \nu$, the denominator does not vanish.

Unlike matching the silhouettes, which are well-separated, matching internal edge

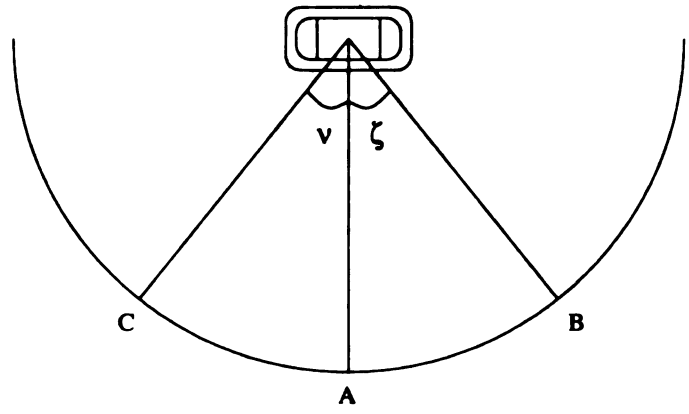


Figure 2.5: Building a model using three images. Points A, B, and C are the three camera locations along a circle in space perpendicular to the Y axis (based on [9]).

points in the images gives rise to the correspondence problem, which is usually tedious and error-prone. However, in our object modeling, the object is either rotated around the X-axis or the Y-axis; thus, the stereo epipolar constraint will force the corresponding points to be on either the same horizontal line or the same vertical line. The rotation center in the X-axis, x_c, can be obtained by *clustering* on all possible triples of edge points (x_1, x_2, x_3) from the same horizontal line with the constraint $x_2 \leq x_1 \leq x_3$ (because x_2, x_1, and x_3 are in the right, central, and left images, respectively). Eq. (2.7) can be introduced as a strong constraint to remove inconsistent (x_1, x_2, x_3) triples where the computed x_c, obtained by substituting x_1, x_2, and x_3 into Eq. (2.7), deviates from the clustered x_c. Using a similar technique, we can also resolve the ambiguity of point correspondence in the Y-axis and compute y_c. Expanding Eq. (2.3), one can notice that the rotation center z_c need not be computed explicitly since the weak perspective projection model is used. Thus, similar to the representation of boundary edges in the model, each internal edge point is associated with the relative depth $z - z_c$ and the rotation center $(x_c, y_c, 0)$.

For each model aspect, the following data is stored. If available, we store the three rotational parameters that relate the local coordinate frame of the aspect to a global coordinate frame for the entire model. The 4 rotation angles, ν_x, ζ_x, ν_y, and ζ_y (as in Figure 2.5, where the subscripts denote the rotation axis), used to create the 4 viewpoints are stored in order to define the limits of the aspect. Notice that x_c and y_c are global to the aspect (and perhaps for the entire model), and thus are stored only once; however, these two parameters are used to generate the predicted image for each internal edge point using Eq. (2.3). For each silhouette point i, we store $(\mathbf{p_i}, \mathbf{r_i})$ where $\mathbf{p_i}$ and $\mathbf{r_i}$ are as in Eq. (2.2). Recall that x_c and y_c are already combined in $\mathbf{r_i}$ for each silhouette point i. Note that we have 3D information for each internal edge point and approximate 3D information for a small neighborhood of each silhouette point. Moreover, because of the construction process, we do not have to do any hidden surface or line removal when using this model aspect to generate an image edgemap.

2.4 Implementation Details

This section presents the implementation details regarding the acquisition of aspect models from 2D images. We describe (1) how the viewsphere is tessellated into aspects, (2) how the tripod is used to simulate the local coordinate system of an aspect, and (3) how the five local images of an aspect are matched to derive the $2\frac{1}{2}$D edgemap of the aspect.

2.4.1 Aspect Model Acquisition

There are a variety of approaches for tessellating the viewsphere, depending on whether they are *object-independent* or *object-dependent.* For object-dependent approaches [68], each object is processed separately, and the tessellation of the viewsphere depends on the complexity of the object. The determination of optimal views for an object is a difficult task. Normally, it is carried out by first tessellating the viewsphere into a fixed number of viewpoints, then integrating views, which share the same object properties, into aspects [28, 36]. For object-independent approaches like ours [27] and many others (*e.g.*, [9]), the viewsphere is tessellated into a fixed number of viewpoints. As a consequence, objects are represented by a fixed number of views. Although this method is straightforward, it suffers from representing simple objects with redundant views. In our approach, the viewsphere is tessellated into latitude bands, each of which is then further divided along longitudinal strips (see Figure 2.6(a)). Each viewpoint defines an aspect. The orientation of a viewpoint is then determined by two angles (ψ, γ) as shown in Figure 2.6(b). Recall that each aspect defines a local coordinate system and is associated with five local training images. This kind of tessellation would allow some training images to be shared by neighboring aspects, reducing the number of images required to model an object. Another advantage is that it is easy to compute the orientation of a viewpoint and consequently the relative orientation between any two viewpoints.

The setup for the acquisition of a model at a viewpoint is given as follows. A tripod as shown in Figure 2.7 is used to obtain the orientation of each viewpoint (aspect) and

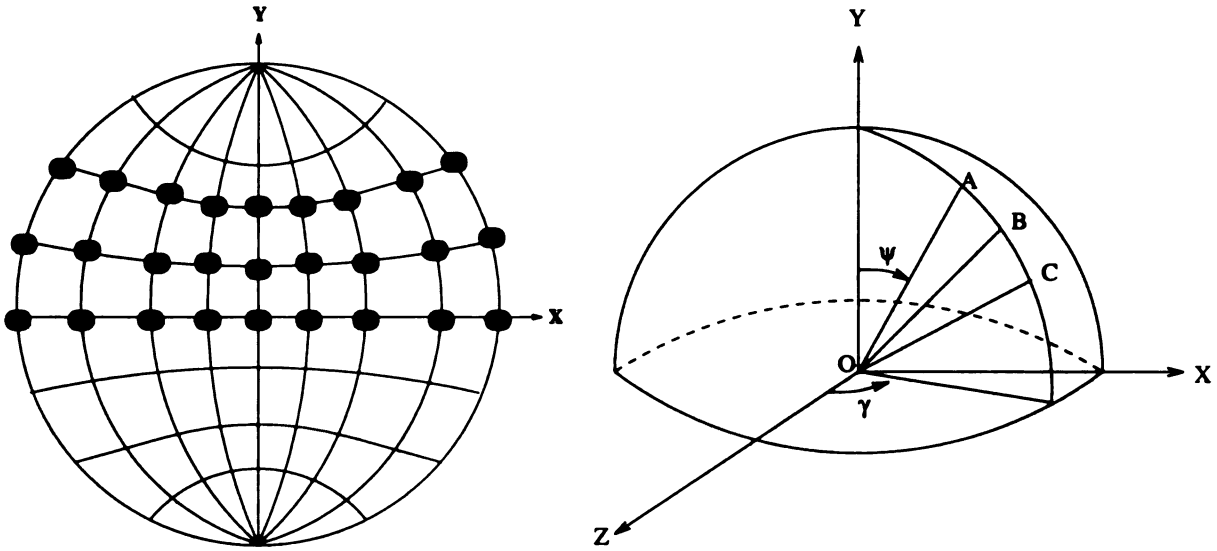


Figure 2.6: The tessellation of the viewing sphere. (a) The sphere is divided into cells by meridians and parallels. (b) Points on the sphere are identified by their longitude ψ and latitude γ.

the five images associated with that aspect. The camera is fixed along the Z-axis with the image plane coinciding with the X-Y plane. The object of interest is mounted on the tripod. The tripod allows two rotations. One is around the X-axis with rotation angle ψ. The other is around the Y-axis with rotation angle γ. The tripod is also mounted on a rotating base which allows one rotation around the Y-axis with rotation angle β. The orientation of a viewpoint is given by the two rotation angles (ψ, γ). After the orientation of a viewpoint is determined, the Cartesian coordinate system, as shown in Figure 2.7, yields the local coordinate system of the viewpoint with ψ and β as the rotation angles around the local X-axis and Y-axis, respectively. These two angles are used to allow the camera to grab the five local images of an aspect.

In our implementation, the following viewpoints specified by (ψ, γ) are selected: $\psi = 40°, 60°, 80°$, and $\gamma = (2n + 1) \cdot 10°$, for $n = 0, \cdots, 17$. For each aspect, both the

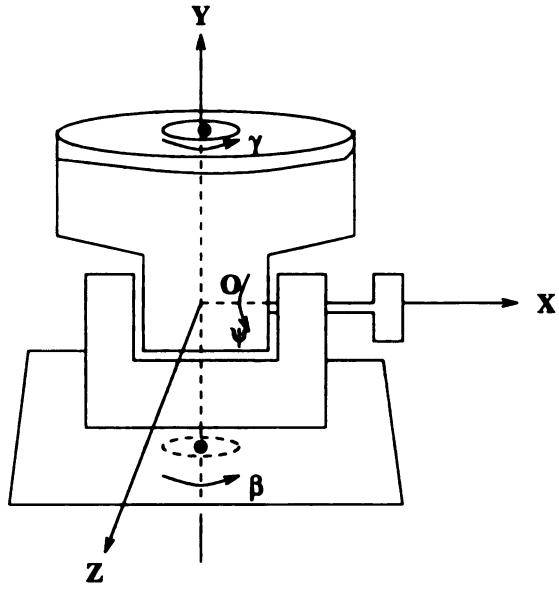


Figure 2.7: The tripod for model construction.

Table 2.1: The number of model aspects of 3D objects.

Object	Number of Aspects	Object	Number of Aspects
camaro	54	phone	18
elephant	54	swan	18
gorilla	54	mug	18
lion	54	blockA	6
pig	54	blockB	6
sharpener	54	truck	6
sprayer	54	can	1
squirrel	54	cup	1
taurus	54	face	1
zebra	54	soap	1

rotation angles around the local X-axis and Y-axis specified by ψ and β, respectively, are $\pm 10°$, giving an aspect breadth of 20°. This yields 54 aspects which cover a band of the northern hemisphere with latitude from 30° to 90°. Table 2.1 lists a set of acquired 3D models (see Figure 2.8) and the number of aspects for each model. Figure 2.9 displays the 54 views of a plastic sprayer. The 54 views in Figure 2.10 are of a plastic lion toy. For each such view, the procedures of the edge detection and the curvature method are applied to derive the $2\frac{1}{2}$D edgemap which is used to predict the edge appearance of an object viewed within the aspect defined by that view. These two procedures are described in the next two sub-sections.

2.4.2 Edgemap Extraction

Extracting the silhouette of an object in an edgemap may be difficult. An object edgemap generated by the Canny edge detector [25] is normally contaminated with noisy edges and background edges. It is rather difficult to extract object edges unless the background is carefully set up when images are taken. The figure-ground separation is a difficult problem. However, in the modeling phase, manual editing of an edgemap is permissible. We use a software program called "CorelPaint" to remove background edges from an edgemap and close the silhouette contour if necessary. After the background edges are removed, the edgemap contains only object edges including silhouette and internal edges. Some of the internal edges touch the silhouette contour, making the automatic separation of these two types of edges difficult. We apply the algorithm described in Figure 2.11 to extract these two types of edges.

(a) camaro (b) elephant (c) gorilla (d) lion

(e) pig (f) sharpener (g) sprayer (h) squirrel

(i) taurus (j) zebra (k) phone (l) swan

(m) mug (n) blockA (o) blockB (p) truck

(q) can (r) cup (s) face (t) soap

Figure 2.8: Examples of model aspects of 3D objects.

Figure 2.9: 54 aspects of a plastic sprayer.

Figure 2.10: 54 aspects of a plastic lion toy.

Input: an edgemap E_{all} containing the silhouette and internal edges of an object.

Output: the silhouette E_s and the internal edges E_i.

Processing:

(1) Find the two extreme edge locations in E_{all} at each horizontal scan line if applicable, and fill in edge points between these two extremes;

(2) Find the two extreme edge locations in E_{all} at each vertical scan line if applicable, and fill in edge points between these two extremes;

(3) "And" the edgemaps generated from (1) and (2);

(4) Locate the boundary of the binary image generated in (3), yielding the object silhouette E_s;

(5) Subtracting E_s from E_{all} gives the internal edges E_i.

Figure 2.11: The edgemap extraction algorithm.

(a) (b) (c)

Figure 2.12: The extracted edgemap of a pencil sharpener. (a) An edgemap containing the internal edges and the silhouette. (b) A binary image resulting from filling edges between boundary pixels. (c) The extracted silhouette from (b). The internal edges can be obtained by subtracting silhouette from the edgemap in (a).

It should be noted that the algorithm specified in Figure 2.11 may not yield the exact silhouette of an object if the object has a non-extreme concave contour segment along the silhouette. In some cases, manual editing of the edgemap has to be performed to ensure that accurate edgemaps are generated. An example of the evolution of an edgemap generated in this manner is illustrated in Figure 2.12.

2.4.3 Matching Model Edge Elements

Once the silhouette and the internal edges of an object are separated, identifying the corresponding points for the silhouette in the pictures is straightforward in the procedure. When the rotation is about the Y-axis, the corresponding points must lie on the epipolar line parallel to the X-axis. Similarly, when the rotation is about the X-axis, the corresponding points must lie on the epipolar line parallel to the Y axis. For each silhouette point, we only need to search in a small range for its corresponding point. Since the silhouette points are well separated, generally there is, in most cases, only one matching candidate to be considered. In the case that there are several candidates, the one with the shortest Euclidean distance is considered as the true match. If the algorithm selects the wrong corresponding point, the outliers will be removed by depth smoothness constraints enforced on the silhouette. Unlike matching the silhouettes, matching internal edge points suffers from the correspondence problem. We apply the heuristic algorithm described in Figure 2.13 to match internal edges.

Four constraints are used to remove the inconsistent corresponding triples of inter-

Input: Five local edgemaps, E_o, E_l, E_r, E_u, and E_d, of an aspect where
E_o is the central image;
E_l and E_r are the left and right images rotated around the X-axis;
E_u and E_d are the up and down images rotated around the Y-axis.

Output: The $2\frac{1}{2}$D edgemap for the aspect.

Processing:

(1) Extract the silhouettes and the internal edges from these five edgemaps (as in Figure 2.11);

(2) For each x_o coordinate of internal edge point in E_o, find all possible pairs of (x_l, x_r) from E_l and E_r which satisfy $x_r < x_o < x_l$;

(3) Derive the x_c coordinate of the rotation center and the relative depth z using Eqs. (2.6) and (2.7);

(4) Histogram on all x_c's and locate the peak $\hat{x_c}$;

(5) Remove all triples (x_l, x_o, x_r) in which the derived x_c deviates from $\hat{x_c}$;

(6) Use all z's from the consistent triples (x_l, x_o, x_r) which survive from step (5) to compute the mean z_μ and the standard deviation z_σ;

(7) Remove all triples (x_l, x_o, x_r) in which the derived z is not in the range of $[z_\mu - 2z_\sigma, z_\mu + 2z_\sigma]$;

(8) Repeat steps (1) to (7) for each y_o coordinate of an internal edge point in E_o to derive y_c and z;

(9) Remove the outliers from each internal edge contour in E_o using depth smoothness constraint.

Figure 2.13: The stereo matching algorithm for modeling internal edges.

nal edges as depicted in steps (2), (5), (7), and (9) of the stereo matching algorithm in Figure 2.13. The first one is the stereo constraint as previously described. The second one is the consistency constraint on the rotation center. Both constraints are very effective in removing inconsistent triples from further consideration. Note that internal edges are located on visible surfaces, the relative depth z of each internal edge point should not differ much from neighbors. The third constraint uses the statistics of the relative depth to remove inconsistent triples. Finally, the depth smoothness constraint, which stipulates that the depth values along the same edge contour should vary smoothly, is used to remove the outliers along each edge contour.

After the spatial coordinate of z, the rotation center (x_c, y_c), and the curvature vector $[r_x, r_y, 0]$ of a model aspect are derived, it is straightforward to apply Eqs. (2.2) and (2.3) to predict the appearance of the model rotated around an arbitrary axis within the aspect. An example is illustrated in Figure 2.14. Figure 2.14(a)-(e) give five local edgemaps of an aspect representing the central, left, right, up, and down views of a car (Ford Taurus), respectively. Figure 2.14(f) is the matched edgemap from these five local edgemaps. Note that some of edge points in this matched edgemap are removed due to inconsistency in the curvature or depth along the contour, or due to the self-occlusion which causes some contours to appear in one edgemap but not in the other. Figure 2.14(g)-(j) are the predicted edgemaps superimposed on the original edgemaps given in Figure 2.14(b)-(e). Since these four edgemaps, as in Figure 2.14(b)-(e), represent the extreme views of the aspect, the results as shown in Figure 2.14(g)-(j) indicate the viability of the modeling scheme.

Table 2.2 lists the maximum (MAX) error and the root mean square (RMS) error

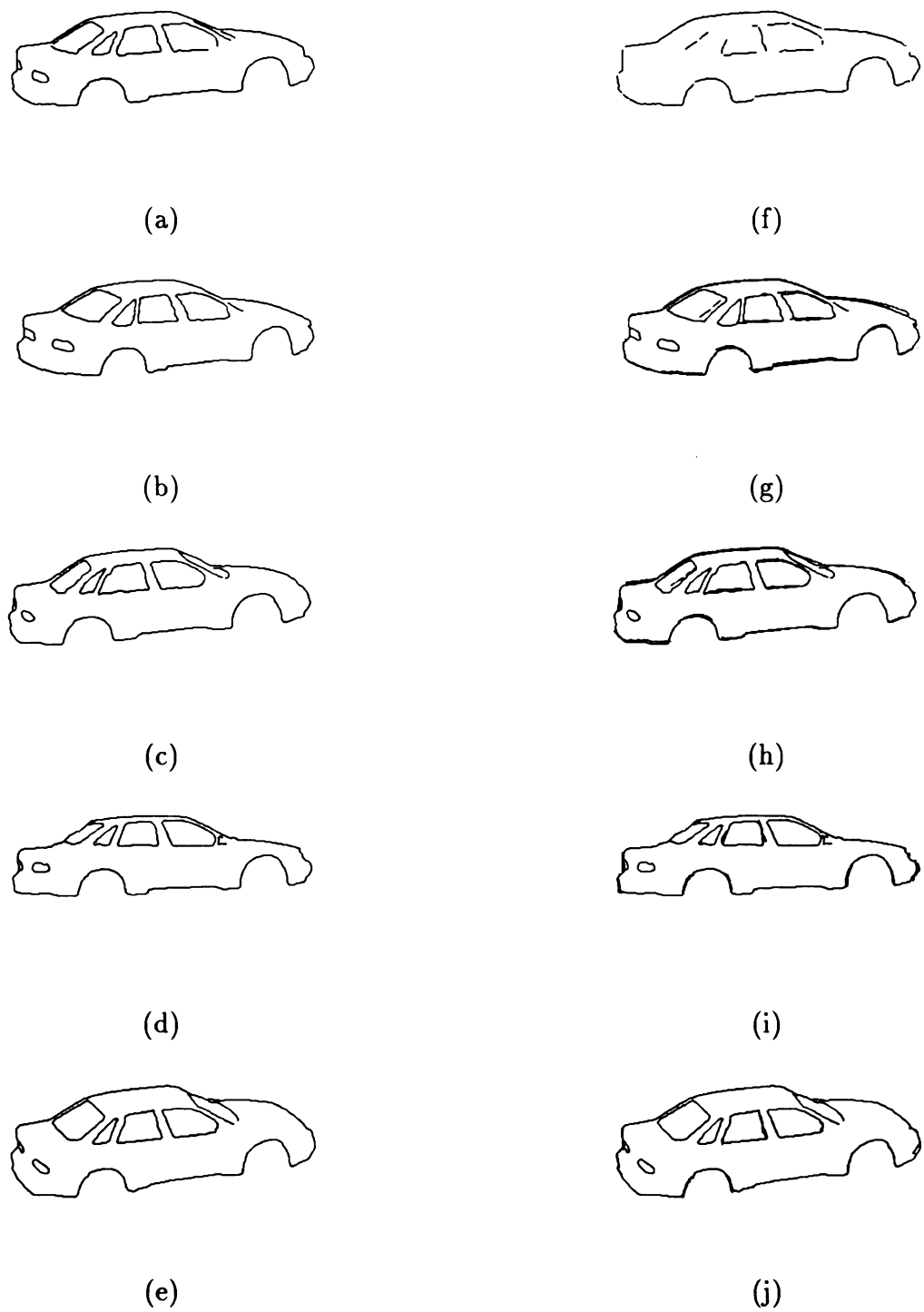


Figure 2.14: The results from the stereo matching algorithm. (a)–(e) The five local edgemaps for the curvature method. (f) The derived $2\frac{1}{2}$D edgemap after the matching. (g)–(j) The predicted edgemaps superimposed on the original edgemaps.

of the alignment of the four views in Figure 2.14. Large MAX and RMS errors in left and right views indicate that there are several outliers in the alignment; the most obvious one is the vertical edge segment near the rear window (see Figure 2.14(g)); since the edge segment is vertical, it has no effect on the up and down views. In general, the prediction of the object's edge appearance is good when the RMS error is small. The outliers will be discarded as occluded edges in the matching process, treated in Chapter 3, and thus they will have little effect on matching. The error analysis of the modeling scheme and more examples of model alignment are given in the next section.

Table 2.2: The prediction error of the vehicle in Figure 2.14.

Model View	MAX Error (pixels)	RMS Error (pixels)
left	7.0	1.78
right	5.0	1.46
up	5.0	0.83
down	4.1	0.68

2.5 Experimental Results

In this section, we are interested in investigating (1) what attributes to the modeling error, and (2) how wide an aspect can be such that the modeling scheme can produce a good approximation of the predicted appearance. We also intend to demonstrate the vitality of the modeling scheme on modeling both polyhedra and sculptured objects. Note that the model alignment in this section involves no projection, scaling, and translation, but only rotation, and the prediction error means the error of alignment

in which only rotation is involved.

Four experiments are reported. The first experiment was conducted with an ellipsoid to validate the theoretical error analysis of the curvature method in [8] using our empirical results. The second and third experiments were conducted with synthetic data generated from a superquadric model to answer the two questions listed above. The fourth experiment with polyhedra and sculptured objects was conducted to analyze the modeling error by aligning models with views from the same aspect.

2.5.1 Error Analysis from an Ellipsoid

The curvature method can predict exactly the appearance of objects with sharp boundaries (for which the radius of curvature is zero) and the appearance of spherical and cylindrical objects (for which the center of the curvature circle coincides with the rotation axis). However, for smooth objects with arbitrary structures, the curvature method only gives an approximation of their predicted appearance. In order to demonstrate the properties of the curvature method, we applied the method to an ellipsoid and analyzed the errors obtained.

To simplify the analysis, consider a canonical ellipsoid $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$ rotating around the vertical (Y) axis. The error depends on the shape of the ellipsoid, that is, the relative length of its axes, and it increases as the ellipsoid becomes elongated in the Z-direction. Let $\mathbf{p} = (x, y)$ be a point on the silhouette of the ellipsoid (refer to Figure 2.4), $\bar{\mathbf{p}} = (\bar{x}, \bar{y})$ the new rim point replacing $\mathbf{p}$ when the ellipsoid is rotated around the Y axis by an angle α, and $\hat{\mathbf{p}} = (\hat{x}, \hat{y})$ the approximated position of $\mathbf{p}$

according to the curvature method. Let $\mathbf{\Delta p} = (\Delta x, \Delta y)$ denote $\hat{\mathbf{p}} - \bar{\mathbf{p}}$. The horizontal section of the ellipsoid through $\mathbf{p}$ is an ellipse centered around $\mathbf{p_o} = (0, y_o)$. Note that $y = \bar{y} = \hat{y} = y_o$ because points $\mathbf{p}$, $\bar{\mathbf{p}}$, $\hat{\mathbf{p}}$, and $\mathbf{p_o}$ all lie on the same horizontal section. The relative error is defined as

$$E = \frac{\|\mathbf{\Delta p}\|}{\|\vec{\mathbf{p_o p}}\|} = \left|\frac{\hat{x} - \bar{x}}{x}\right| \tag{2.8}$$

This error, as derived in [8], is given by

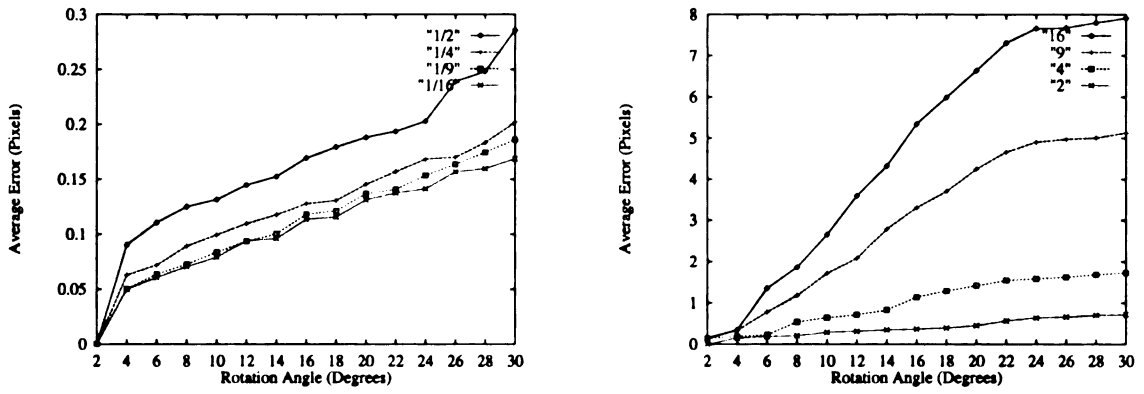
$$E(\frac{c^2}{a^2}, \alpha) = \cos\alpha + \frac{c^2}{a^2}(1 - \cos\alpha) - \sqrt{\cos^2\alpha + \frac{c^2}{a^2}\sin^2\alpha}. \tag{2.9}$$

In the error analysis of alignment, we are interested in the absolute error defined as

$$E' = \|\mathbf{\Delta p}\| = \|\vec{\mathbf{p_o p}}\| \cdot E \tag{2.10}$$

The error in Eq.(2.10) depends only on three parameters, the length of the vector $\vec{\mathbf{p_o p}}$, i.e., $\|\vec{\mathbf{p_o p}}\|$, the rotation angle α, and the aspect ratio of the ellipsoid $\frac{c^2}{a^2}$.

To analyze the error of the predicted appearance, we used a superquadric model to generate eight ellipsoids with aspect ratio ($\frac{c^2}{a^2}$) of $\frac{1}{16}$, $\frac{1}{9}$, $\frac{1}{4}$, $\frac{1}{2}$, 2, 4, 9, and 16, respectively. The aspect breadth was set to 90°. For each aspect ratio, fifteen test data were generated, each varied by rotating 2 degrees about the Y axis. For each ellipsoid, we compute the alignment error, i.e., the absolute error E' in Eq. (2.10). Note that for the simplicity of the analysis, the value of a was fixed and only the value of c was varied when each ellipsoid was generated. Thus, $\|\vec{\mathbf{p_o p}}\| = a$ because $\mathbf{p}$



(a) (b)

Figure 2.15: The error of the curvature method as a function of the aspect ratio $\frac{c^2}{a^2}$ and the rotation angle α. (a) $\frac{c^2}{a^2} = \frac{1}{16}$, $\frac{1}{9}$, $\frac{1}{4}$, and $\frac{1}{2}$. (b) $\frac{c^2}{a^2} = 2$, 4, 9, and 16.

is a point on the silhouette of the ellipsoid before any rotation, implying $E' \propto E$. As shown in Figure 2.15, the error behaves differently in each of the two ranges: (1) when $c \leq a$, and (2) when $c > a$. In the first case, the ellipsoid's width is larger than its depth, the error assumes small values even for fairly large values of α. In the second case, the ellipsoid's depth is larger than its width, the error assumes larger values even for fairly small values of α. As a function of α, the error function is symmetric and the absolute value of the error increases monotonically with the absolute value of α. Figure 2.15 also depicts this phenomenon. All these empirical results are consistent with the theoretical results given in [8].

2.5.2 Rotation Axis Effect

The curvature method uses five local images of an object rotated around the X axis and the Y axis within an aspect to predict the appearance of the object after a rotation. Rotating an object about an arbitrary axis (not coincident with the X axis

or the Y axis) creates larger error than rotating the object about either the X axis or the Y axis. This is because the former involves both the curvature radii r_x and r_y while the latter only involves either r_x or r_y. Note that the rotation axis N is parameterized by $(\cos\theta\sin\phi, \sin\theta\sin\phi, \cos\phi)$. Let $Nk, k = 0, \cdots, 9$ denote the 10 rotation axes with $\phi = 90°$ and $\theta = k \cdot 10°$. Then $N0$ and $N9$ correspond to the X axis and the Y axis, respectively. Figure 2.16(a)-(e) show the five local images of a model aspect with an aspect breadth of 40°. In this experiment, we generated 70 test data by rotating the aspect model (see Figure 2.16(a)) around these 10 rotation axes with rotation angles, varied from 2° to 14°, each differing by 2° and denoted as *rot2*, *rot4 rot6*, *rot8*, *rot10*, *rot12*, and *rot14*, respectively, in Figure 2.17. Each predicted appearance of the model was aligned with the actual appearance of the model to compute the prediction error of the curvature method. As the results show in Figure 2.17, the prediction error tends to be larger with larger deviation of the rotation axis from the two main axes, and also the prediction error becomes larger as the rotation angle increases. Thus, we have to tolerate a few pixels of deviation when model rotation angles are extreme.

2.5.3 Aspect Breadth Effect

As mentioned in the previous section, the viewsphere may be tessellated into a fixed number of viewpoints. This immediately leads to a question of how wide of an aspect is appropriate. For objects with sculptured surfaces, the tessellation has to be as local as possible to have better approximation of the predicted appearance. Wide aspects

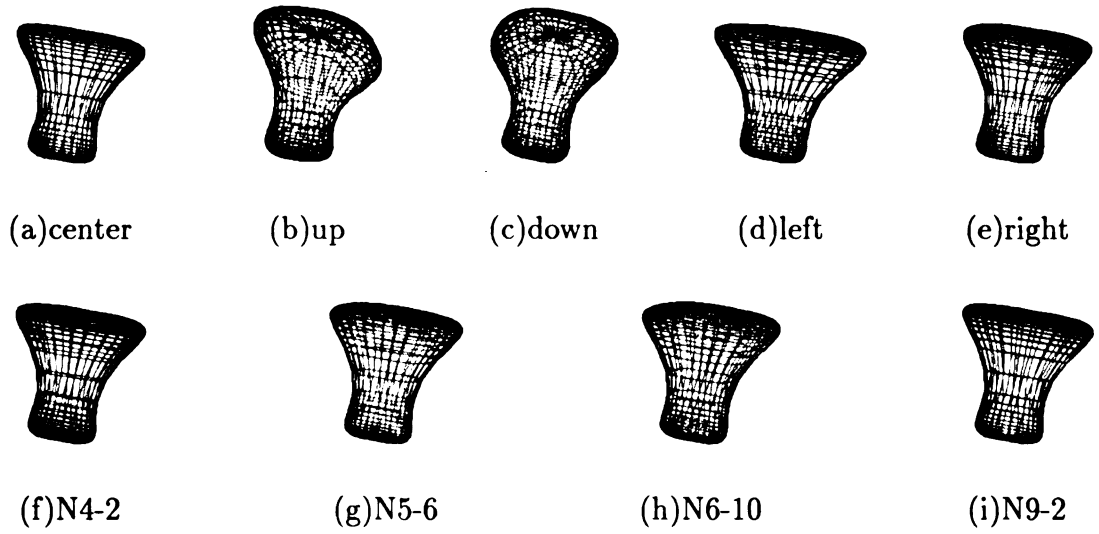


Figure 2.16: An aspect of a superquadric model and its four test data. (a)-(e) The five images of a model aspect with aspect breadth of 40°. (f)-(i) The test data in the model aspect.

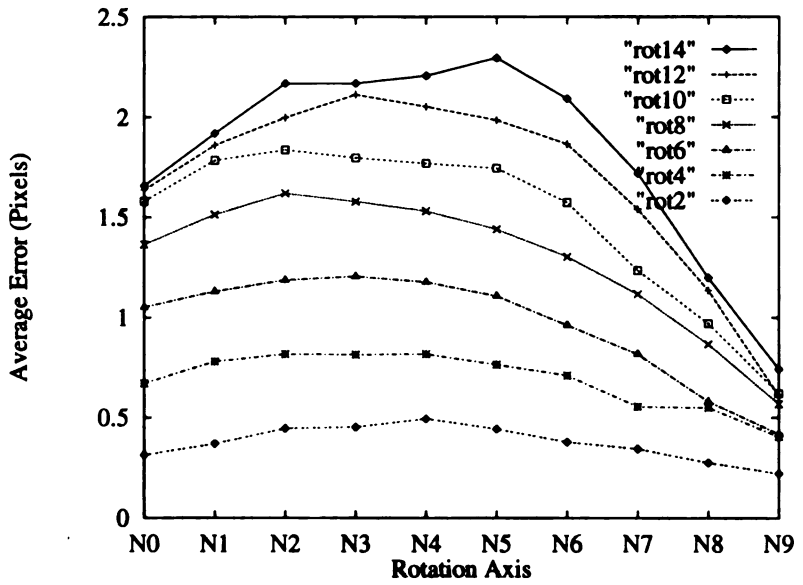


Figure 2.17: The error of the curvature method as a function of the rotation axis and the rotation angle for the object in Figure 2.16(a).

increase the chance of self-occlusion, and consequently degrade the approximation of the predicted appearance. On the other hand, narrow aspects may represent some simple objects with redundant views, and thus, require a large amount of storage, but the accuracy of the predicted appearance is increased accordingly.

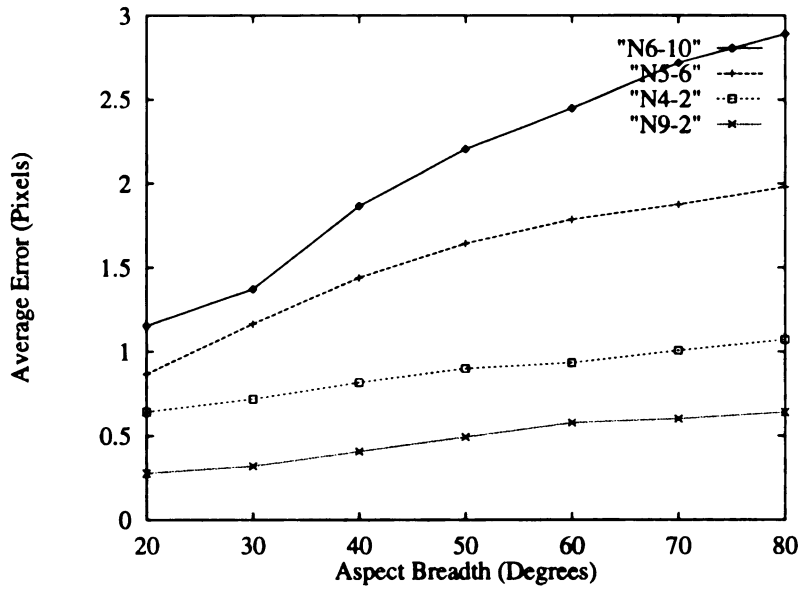


Figure 2.18: The error of the curvature method as a function of aspect breadth for the object in Figure 2.16(a).

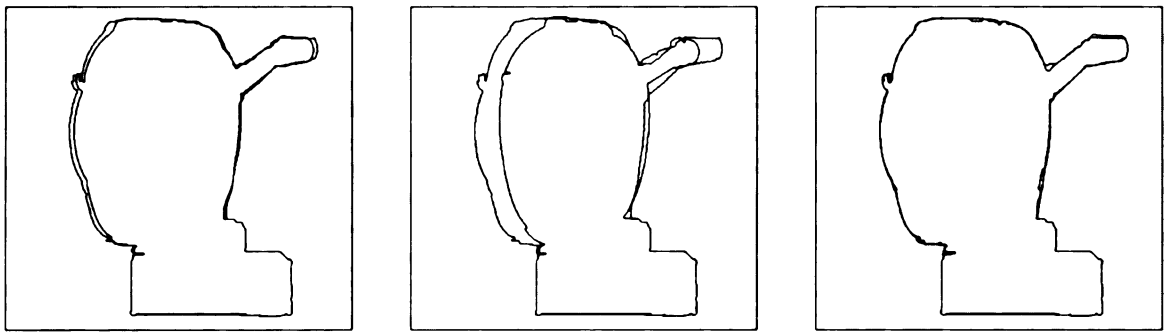
In order to investigate the effect of aspect breadth, seven aspects of a superquadric model were constructed with aspect breadth varied from 20° to 80°. Figure 2.16 (a)-(e) show one of the model aspects with aspect breadth of 40°. Figure 2.16(f)-(i) show four sets of generated test data, $N4-2$, $N5-6$, $N6-10$, and $N9-2$, representing the object being rotated around the rotation axes $N4$, $N5$, $N6$, and $N9$ with rotation angles of 2°, 6° 10°, and 2°, respectively. Then, for each aspect, the curvature method was applied to the 4 test data. The prediction error is shown in Figure 2.18; the modeling error increases monotonically as the aspect breadth becomes larger. These

empirical results suggest that a smaller aspect breadth be adopted for tessellating the viewsphere if a large rotation is required to predict the appearance of an object. Also, a smaller aspect breadth makes the prediction more accurate, and consequently, makes the matching (model fitting), treated in Chapter 3, more efficient in the sense that the fitting would not wander around in parameter space due to ambiguous matches.

2.5.4 Model Alignment

To demonstrate the vitality of the curvature method, we chose twenty models for alignment in which only rotation was involved. All objects were observed under the weak perspective viewing model. Figure 2.19 shows the results of aligning the predicted object model with the images. It can be seen in Figure 2.19(a) and (b) that rotations create large deformations of the silhouettes. Figure 2.19(c) gives an accurate alignment of the predicted object model with its corresponding sensed image. A simple distance metric between the image edgemap and the aligned model appears to be sufficient to select the correct model.

Figures 2.20 and 2.21 illustrate the alignments of twenty predicted models with their corresponding sensed images. Note that a model is specified by the model name followed by an aspect number. For example, *blockA1* represents aspect 1 of the *blockA* model. For models with only one aspect, the aspect number is not specified. The white contours represent the predicted models. Those isolated white edge points (see Figure 2.20(a) and (b) for larger pictures) indicate the outliers caused by the following two effects: (1) the five training images were not well cleaned; and (2) the matching



(a) (b) (c)

Figure 2.19: Deformation of the images with respect to the predicted model. (a) A deformation of the pencil sharpener after a rotation of $-10°$ around the vertical axis. (b) A deformation of the pencil sharpener after a rotation of $+20°$ around the vertical axis. (c) Alignment of the pencil sharpener model with its corresponding sensed image.

algorithm selected the wrong corresponding points. These two problems can be solved by spending more time in editing and analyzing the five training images or the $2\frac{1}{2}$D edgemap derived from the five training images. Table 2.3 lists results of aligning these twenty models with their corresponding sensed images. The average alignment errors for all objects are within 1.0 pixel. It should be noted that accurate predictions were achieved despite the fact that (1) the object had complex 3D shape; (2) the light reflectance distorted some of the internal edges; and (3) only five images were used to create a crude approximation of the radii of curvature.

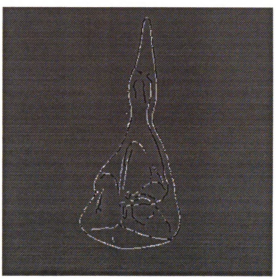
2.6 Summary

In this chapter, we have presented a method to construct an aspect model for a complex rigid object. We have studied and implemented the curvature method proposed by Basri and Ullman to predict the new appearance of an object with smooth surface

Table 2.3: Experimental Results of Model Alignment.

Model	Rotation Axis $[n_x, n_y, n_z]$	Rotation angle	Avg. Error (pixels)
blockA1	[1,0,0]	+10°	0.90
blockB1	[0,1,0]	+10°	0.77
camaro8	[1,0,0]	-10°	0.62
can	[0,1,0]	+10°	0.56
cup	[0,1,0]	+10°	0.29
elephant1	[0,1,0]	+10°	0.29
face	[1,0,0]	-10°	0.77
gorilla1	[0,1,0]	+10°	0.51
lion10	[1,0,0]	+10°	0.79
mug7	[0,1,0]	-10°	0.60
phone1	[1,0,0]	-10°	0.43
pig1	[0,1,0]	+10°	0.81
sharpener10	[0,1,0]	+10°	0.83
soap	[1,0,0]	-10°	0.26
sprayer1	[0,1,0]	+10°	0.46
squirrel19	[1,0,0]	+10°	0.95
swan2	[1,0,0]	-10°	0.69
taurus20	[0,1,0]	+10°	0.43
truck1	[1,0,0]	+10°	0.60
zebra1	[0,1,0]	+10°	0.50

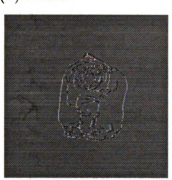
(a) phone1



(b) swan2

(c) camaro8

(d) elephant1



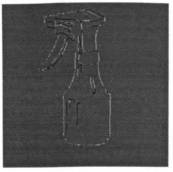
(e) gorilla1

(f) lion10

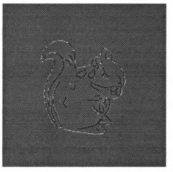
(g) pig1

(h) sharpener10

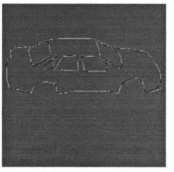
Figure 2.20: Examples of model alignment. The white contour is the predicted model superimposed on the corresponding sensed image.



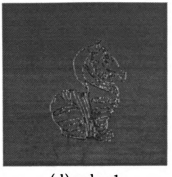
(a) sprayer1



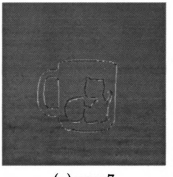
(b) squirrel19



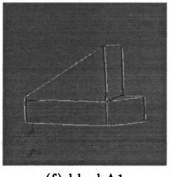
(c) taurus20



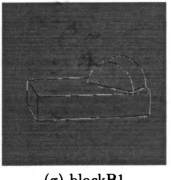
(d) zebra1



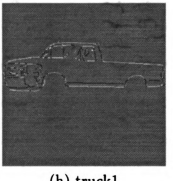
(e) mug7



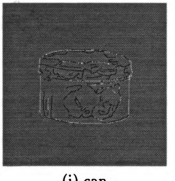
(f) blockA1



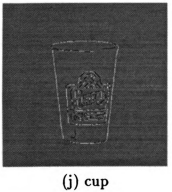
(g) blockB1



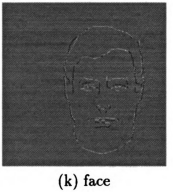
(h) truck1



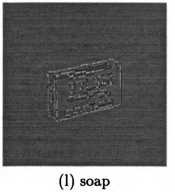
(i) can



(j) cup



(k) face



(l) soap

Figure 2.21: More examples of model alignment. The white contour is the predicted model superimposed on the corresponding sensed image.

following a 3D rotation. A model aspect is constructed from 5 intensity images and alignment is done with a single intensity image. We have shown that three pictures are in principle sufficient for approximating the radius of a curvature r_x or r_y, and five pictures can be used to estimate the components r_x and r_y independently. If both components r_x and r_y are present, the new appearance of an object following a rotation about an arbitrary axis can be predicted.

The experimental results are supportive of our design. The proposed modeling scheme was shown vital in modeling both polyhedra and sculptured objects. It was also found to give accurate results for large transformations. In this scheme, an object is represented using a number of viewer-centered model aspects, rather than a single object-centered geometric model. Each model aspect covers a range of many possible neighboring viewing angles. To represent the edge appearance of an object from any viewpoint, a number of model aspects are required. The computations required in this scheme during the prediction stage are simple; for example, no hidden line removal is necessary. Our model construction is almost automatic when object prototypes are available. Our work both validates and extends the work of Basri and Ullman [9].

Chapter 3

Matching 3D Object Models to 2D Images

3.1 Introduction

In this chapter, we address the problem of matching 3D objects with arbitrary curved surfaces to a single 2D intensity image, i.e., the object localization problem. We take a general viewpoint, but we are interested in specific applications as well, such as detecting the pose of industrial objects or bones in X-ray images. The work fits within the theoretical framework for object recognition via alignment espoused by Ullman [111] and Lowe [76]. Four stages are involved in this framework: (1) Model building, as discussed in Chapter 2, for representing 3D objects; (2) image processing, including segmentation, extracts object features (typically edge points); (3) configurations of features are used to index to candidate models and candidate poses; and (4) determination of pose parameters that best align the projected 3D models and the observed

image features in order to verify object presence. Stages 1 and 2 were presented in Chapter 2, and stage 3 will be treated in Chapter 4: in this chapter, we present contributions to stage 4, that is, object localization via alignment. A matching method is developed which can handle objects with partial occlusion and which also includes internal object edges, in addition to object silhouette, in the matching process for a more accurate pose estimation. Matching does not assume the existence of salient local features in the image and hence is directly applicable to smooth objects. During matching, a heuristic is adopted to synthesize correspondences in order to refine the pose parameters for diminishing the matching error between the model and observed edgemaps.

We assume that (1) objects are rigid; (2) the object model used for pose estimation is known; and (3) an approximate orientation of the object in the image is also given. Assumptions 2 and 3 can be satisfied by the indexing scheme presented in Chapter 4 because the indexing scheme will provide candidate object models and also approximate particular views of the object models. The object may be partially occluded. Our program takes as input an object view roughly 20 degrees in breadth, an object image, and a match tolerance. It outputs a refined object pose when matching converges and otherwise reports failure. In our experiments and core implementation, we have implicitly assumed that the object is not rotated around the visual axis, i.e., a normalized zero-roll angle for the object as discussed in Chapter 2. A rough alignment of the image plane based on object features and image features will be discussed in Chapter 4. This assumption fixes one degree of freedom of the rotation and has, no doubt, contributed to the performance reported. However, the method presented

applies to the general weak perspective case with six free parameters: one for scaling, two for translation, and three for rotation.

Based on the assumptions, we define the problem more formally as follows.

Problem Statement: Let $\mathcal{T}$ be a set of transformations, including rotations in 3-space, translations and scale changes, followed by an orthographic projection. Let $\mathcal{E}_M = \{E_1, E_2, \cdots E_n\}$ be a set of view-centered $2\frac{1}{2}$D edgemaps representing a 3D aspect model M. Given an edgemap E_o of an observed object in a scene, an aligning transformation $T \in \mathcal{T}$ is sought such that the predicted edgemap of M, denoted as E_m, generated by the application of T matches E_o.

This chapter is organized with the next section reviewing some existing techniques for matching objects with sculptured surfaces. Section 3.3 presents the proposed method for computing pose via matching a model to an observed image. An analysis of the sensitivity of pose parameters to object shape and a study of the basin of convergence are also presented in this section. Section 3.4 shows experimental results. The last section summarizes the proposed approach.

3.2 Related Background

We are interested in locating 3D objects with sculptured surfaces from single 2D intensity images. Several previous approaches to recovery of object pose from a single image under perspective or weak perspective projection require that three or more corresponding feature points be identified in both the object model and the sensed image [41, 60, 76]. Such points are identified by salient features which are usually de-

pendent on the type of object and hence both the model and feature extraction need to be specialized. The silhouette of a 3D object has also been used in object modeling [9, 113, 121] and pose estimation [69, 107]. However, for a smooth object, the rim, which we perceive as a silhouette, projected into an image moves and deforms over the 3D surface according to the viewpoint position; thus, identification of point correspondences is frustrated by lack of salient and precise local features. Furthermore, the silhouette alone does not provide for accurate location of smooth and somewhat symmetric objects because the object silhouettes from different viewpoints often appear similar. The lack of sensitivity of the silhouette to the pose parameters not only precludes accurate pose estimation, but also causes wandering and slow convergence in hill-climbing techniques. For this reason, we are forced to introduce the use of any available *internal object edges* resulting from surface marks or creases to help refine object pose.

There have been efforts that have succeeded in locating smooth 3D objects from their 2D image contours. Lowe [77] devised an iterative approach based on Newton's method for solving for projection and model parameters that best fit a smooth 3D object to matching 2D image contours. In his approach, the object is approximated by piecewise polygonal surface patches which may be derived from a CAD model. Kriegman and Ponce [69] used an elimination method to construct an implicit equation for the object image contours, and then determined the parameters of the implicit equation by reducing it to a fitting problem between the theoretical contour and the observed data. This approach requires a precise geometric model of the object but is very general regarding the type of features which may be used. Sullivan *et al.* [107]

approached the problem of pose estimation from image contours by using a combination of constrained optimization and nonlinear least-squares estimation techniques to minimize the mean-squared *geometric* distance between the viewing cone and a parameterized surface. Huttenlocher *et al.* [59] used the Hausdorff metric to correlate contour points of a model to those of an observed object. This method does not require point correspondences and can handle partially occluded or distorted objects. While this method is attractive for handling 2D translation, like many other methods (*e.g.*, [49, 55, 60, 76, 85]), including ours, its applicability to 2D rotations and 3D transformations suffers from the need to search over many possibilities.

The proposed matching scheme has been used in our previous work [27] to locate smooth objects based on the silhouettes. We were able to locate partially occluded objects in a scene and efficiently track a moving object from a sequence of images. While translation and scaling parameters were accurately recovered, rotation parameters could not be precisely obtained. In this chapter, we extend the previous method to include the use of internal object edges to increase recognition accuracy and to permit more accurate computation of pose. Significant updating of both the modeling and matching schemes was required. In the modeling scheme, as presented in Chapter 2, the curvature method is augmented with the internal edges. In the matching scheme, internal edges, in addition to silhouette, are used in the fitting process where a heuristic is adopted to ensure that the parameter fitting is not overly biased to either internal or boundary edges. In addition, we show results of a study of the convergence of the matching method over the many viewpoints of an aspect. Our convergence results are general and lend support to other techniques based on fitting

[69, 77, 89, 107]. Our approach differs from the aforementioned approaches in several aspects. First, objects are with sculptured smooth surfaces in which no piecewise approximation or parametric model is used to describe object surfaces. Second, no salient local features are used as matching primitives. Third, an efficient 2D representation of a 3D object is used.

3.3 Matching Model Edgemaps to Observed Edgemaps

In Chapter 2, we have presented the curvature method for recovering an object's approximate appearance given a number of view-centered 2D projections. In this section, we intend to show how to align model data to image data for computing object pose and confirming object presence. We want our method to be able to tolerate image processing errors, image artifacts, and the presence of other objects which may partially occlude the object of interest. Recall that from Chapter 2, an aspect model is constructed as $\bar{\mathbf{p}} = \mathbf{R}(\mathbf{p} - \mathbf{r}) + \mathbf{r}$ where $\mathbf{r} = (r_x, r_y, 0)^t$ is the radius of curvature for rim points and $\mathbf{r} = (x_c, y_c, 0)^t$ for internal edge points, and $\bar{\mathbf{p}}$ is the new predicted point for $\mathbf{p}$. Once an aspect model is constructed as above, it is simply a matter of transforming and projecting new rim points and internal edge points $\bar{\mathbf{p}}$ to generate a predicted 2D edgemap. Under weak perspective projection, the rim point or internal edge point $\bar{\mathbf{p}}$ on the model object edgemap is given in the observed object

camera coordinate system by:

$$\hat{\mathbf{p}} = \bar{\mathbf{p}} + \mathbf{t}. \tag{3.1}$$

where $\mathbf{t}$ is a translation between origins of model and image coordinate systems. The image coordinates (u, v) of an edge point $\hat{\mathbf{p}} = (\hat{x}, \hat{y}, \hat{z})^t$ on the edgemap of the model object are given by

$$(u, v) = (s\hat{x}, s\hat{y}) \tag{3.2}$$

where s is a scale factor. Note that the Z coordinate of $\mathbf{t}$ has no effect on the resulting image coordinates and can be set to 0 or ignored.

Combining Eqs. (3.1) and (3.2) with the curvature method, we obtain a transformation as follows:

$$\mathbf{q} = \mathbf{S}[\mathbf{R}(\mathbf{p} - \mathbf{r}) + \mathbf{r} + \mathbf{t}] \tag{3.3}$$

where $\mathbf{q} = (u, v, 0)^t$, $\mathbf{t} = (t_x, t_y, 0)^t$, $\mathbf{S}$ is the weak perspective projection matrix with a scale factor s, and $\mathbf{R}$ is the rotation matrix with rotation angle α, and rotation axis $\vec{\mathbf{N}} = (n_1, n_2, n_3)$. We re-parameterize the rotation axis using the longitude θ and the latitude ϕ of the unit sphere: $\vec{\mathbf{N}} = (n_1(\theta, \phi), n_2(\theta, \phi), n_3(\theta, \phi)) = (\cos\theta \sin\phi, \sin\theta \sin\phi, \cos\phi)$. The specific components of these two matrices are given by:

$$\mathbf{S} = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 0 \end{bmatrix};$$

$$\mathbf{R} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}$$

$$= \begin{bmatrix} (1-n_1^2)\cos\alpha + n_1^2 & n_1 n_2(1-\cos\alpha) - n_3\sin\alpha & n_1 n_3(1-\cos\alpha) + n_2\sin\alpha \\ n_1 n_2(1-\cos\alpha) + n_3\sin\alpha & (1-n_2^2)\cos\alpha + n_2^2 & n_2 n_3(1-\cos\alpha) - n_1\sin\alpha \\ n_1 n_3(1-\cos\alpha) - n_2\sin\alpha & n_2 n_3(1-\cos\alpha) + n_1\sin\alpha & (1-n_3^2)\cos\alpha + n_3^2 \end{bmatrix}$$

Object pose is thus modeled by 5 parameters; two for translation and 3 for orientation; the 6th parameter, the scale factor s, is associated with both the projection and object size; we combine it with the pose parameters in our discussion below.

3.3.1 Newton's Method and Least-Squares Minimization

The transformation in our 3D-to-2D alignment is a nonlinear operation: the best-fit parameters of the transformation can be obtained through a minimization technique which uses a *merit function* to measure the goodness-of-fit. Newton's method is a promising candidate for searching for the best-fit parameters. As is the case with any minimization technique, this method requires an appropriate initial guess of the parameters to prevent from getting stuck in the basin of a local minimum. If the aspect model is viable, we should be able to use any viewpoint of the aspect as an initial guess, and thus initialization will come from indexing. We use the Levenberg-Marquardt method, together with random walks to escape local minima [5, 50].

Although we do not assume any special feature points along the object boundary or the internal edges, our method does extract a number of points where the matching

between object and model edgemaps is the worst. Newton's method will diminish the error between such point pairs if the correct aspect model is used. Our pose estimation task is: given a number of corresponding silhouette points and internal edge points identified in both the model object and the observed object, solve for $\mathbf{t}$, s, and $\mathbf{R}$ (6 unknowns) to drive down the sizes of the worst gaps between the matching edgemaps.

Let $D(E_o, E_m(\vec{\omega}))$ be the mean-squared distance between *certain* points of the observed edgemap E_o and the model edgemap E_m, where $\vec{\omega} = (t_x, t_y, s, \alpha, \theta, \phi)$ is the vector of pose parameters. In the minimization scheme, $D(E_o, E_m(\vec{\omega}))$ serves as the merit function for searching for the best-fit pose parameters $\vec{\omega}$. Let $\mathbf{p_i} = (x_i, y_i, z_i), i = 1, 2, \cdots, N$, be a set of 3D points of the model m, and let (u_i, v_i) be a set of 2D image points of the observed edgemap E_o. Recall that the z_i is the one modified by z_c as discussed in Section 2.3. Let $(F_u(\mathbf{p_i}; \vec{\omega}), F_v(\mathbf{p_i}; \vec{\omega}))$ be the aligning transformation as defined in Eq. (3.3), i.e.

$$\begin{aligned} F_u(\mathbf{p_i}; \vec{\omega}) &= s[R_{11}(x_i - r_x) + R_{12}(y_i - r_y) + R_{13}z_i + r_x + t_x], \\ F_v(\mathbf{p_i}; \vec{\omega}) &= s[R_{21}(x_i - r_x) + R_{22}(y_i - r_y) + R_{23}z_i + r_y + t_y] \end{aligned}$$

where R_{ij}'s are the components of the rotation matrix $\mathbf{R}$.

Our goal is to minimize the mean-squared distance

$$D(E_o, E_m(\vec{\omega})) = \frac{1}{N}\sum_{i=1}^{N}\{(F_u(\mathbf{p_i}; \vec{\omega}) - u_i)^2 + (F_v(\mathbf{p_i}; \vec{\omega}) - v_i)^2\}. \qquad (3.4)$$

The solution to Eq. (3.4) can be obtained iteratively by using Newton's method:

$$\vec{\omega}^{(k+1)} = \vec{\omega}^{(k)} - \mathbf{H}^{-1} \cdot [\nabla(\vec{\omega}^{(k)})] \tag{3.5}$$

where $\vec{\omega}^{(k)}$ is the parameter estimate at iteration k, $\mathbf{H}$ is the second derivative matrix (*Hessian* matrix) of D with respect to the parameters $\vec{\omega}$ given by

$$\begin{aligned}\frac{\partial^2 D}{\partial\omega_k\partial\omega_l} &= \frac{2}{N}\sum_{i=1}^{N}\{\frac{\partial^2 F_u(\mathbf{p_i};\vec{\omega})}{\partial\omega_k\partial\omega_l}(F_u(\mathbf{p_i};\vec{\omega}) - u_i) + \frac{\partial F_u(\mathbf{p_i};\vec{\omega})}{\partial\omega_k}\cdot\frac{\partial F_u(\mathbf{p_i};\vec{\omega})}{\partial\omega_l} \\ &+ \frac{\partial^2 F_v(\mathbf{p_i};\vec{\omega})}{\partial\omega_k\partial\omega_l}(F_v(\mathbf{p_i};\vec{\omega}) - v_i) + \frac{\partial F_v(\mathbf{p_i};\vec{\omega})}{\partial\omega_k}\cdot\frac{\partial F_v(\mathbf{p_i};\vec{\omega})}{\partial\omega_l}\}, \quad k,l = 1,\cdots,6\end{aligned}$$

and $[\nabla(\vec{\omega})]$ is the *gradient* of D with respect to the parameter $\vec{\omega}$ given by

$$\frac{\partial D}{\partial\omega_k} = \frac{2}{N}\sum_{i=1}^{N}\{\frac{\partial F_u(\mathbf{p_i};\vec{\omega})}{\partial\omega_k}(F_u(\mathbf{p_i};\vec{\omega}) - u_i) + \frac{\partial F_v(\mathbf{p_i};\vec{\omega})}{\partial\omega_k}(F_v(\mathbf{p_i};\vec{\omega}) - v_i)\}, \quad k = 1,\cdots,6.$$

One of the most expensive aspects of implementing Newton's method is often the computation of the elements of the Hessian matrix. However, *the calculation of partial derivatives in closed form is straightforward* given the aligning transformation: $F_u(\mathbf{p_i};\vec{\omega})$ and $F_v(\mathbf{p_i};\vec{\omega})$. The partial derivatives of F_u and F_v with respect to t_x, t_y and s are trivial. However, it is difficult to calculate the partial derivatives of F_u and F_v with respect to $\mathbf{R}$ since there is no standard formulation of $\mathbf{R}$ in terms of its three parameters. Most researchers represent $\mathbf{R}$ in terms of three Euler angles: yaw, pitch, and roll. The partial derivatives of Cartesian coordinates with respect to these parameters are strikingly simple as mentioned in [76]. However, we are

Table 3.1: The partial derivatives of $\vec{\mathbf{N}}$ with respect to θ, and ϕ.

	n_1	n_2	n_3
θ	$-n_2$	n_1	0
ϕ	$\frac{n_1 n_3}{\sqrt{n_1^2+n_2^2}}$	$\frac{n_2 n_3}{\sqrt{n_1^2+n_2^2}}$	$-\sqrt{n_1^2+n_2^2}$

more interested in finding the rotation angle α and the direction of the rotation axis $\vec{N}$ because $\vec{N}$ gives the normal of the ground plane and α may be used later to derive the motion (trajectory) of the object. Given the parameterization of $\vec{\mathbf{N}}$ as previously described, it is straightforward to derive the partial derivatives of n_1, n_2 and n_3 with respect to their two parameters: θ and ϕ. Table 3.1 gives these partial derivatives for all combinations of n_1, n_2, n_3, θ and ϕ where $\phi \neq k\pi$, $\forall k = 0, 1, \cdots$. When $\phi = k\pi$, $\forall k = 0, 1, \cdots$, $\partial\vec{\mathbf{N}}/\partial\phi = (\cos\theta\cos\phi, \sin\theta\cos\phi, 0)$.

Now we can build up the partial derivatives of R with respect to θ, ϕ and α based on the partial derivatives on Table 3.1. For example,

$$\frac{\partial R_{11}}{\partial\theta} = (1-\cos\alpha)\frac{\partial n_1^2}{\partial\theta} = (1-\cos\alpha)2n_1\frac{\partial n_1}{\partial\theta} = -2n_1n_2(1-\cos\alpha)$$

and

$$\frac{\partial R_{11}}{\partial\phi} = (1-\cos\alpha)\frac{\partial n_1^2}{\partial\phi} = (1-\cos\alpha)2n_1\frac{\partial n_1}{\partial\phi} = \frac{2n_1^2n_3(1-\cos\alpha)}{\sqrt{n_1^2+n_2^2}}.$$

The partial derivatives for all combinations of $\theta, \phi, \alpha, R_{11}, R_{12}, R_{13}, R_{21}, R_{22}$ and R_{23} are listed in Table 3.2.

Given the parameterization F_u and F_v and all partial derivatives of R with respect to θ, ϕ and α, we can accomplish our goal of computing the Hessian matrix

Table 3.2: The partial derivatives of $\mathbf{R}$ with respect to α, θ and ϕ.

	α	θ	ϕ
R_{11}	$(n_1^2-1)S_\alpha$	$-2n_1n_2(1-C_\alpha)$	$2n_1^2n_3(1-C_\alpha)/A$
R_{12}	$n_1n_2S_\alpha-n_3C_\alpha$	$(n_1^2-n_2^2)(1-C_\alpha)$	$(2n_1n_2n_3(1-C_\alpha)+(n_1^2+n_2^2)S_\alpha)/A$
R_{13}	$n_1n_3S_\alpha+n_2C_\alpha$	$-n_2n_3(1-C_\alpha)+n_1S_\alpha$	$(n_1(2n_3^2-1)(1-C_\alpha)+n_2n_3S_\alpha)/A$
R_{21}	$n_1n_2S_\alpha+n_3C_\alpha$	$(n_1^2-n_2^2)(1-C_\alpha)$	$(2n_1n_2n_3(1-C_\alpha)-(n_1^2+n_2^2)S_\alpha)/A$
R_{22}	$(n_2^2-1)S_\alpha$	$2n_1n_2(1-C_\alpha)$	$2n_2^2n_3(1-C_\alpha)/A$
R_{23}	$n_2n_3S_\alpha-n_1C_\alpha$	$n_1n_3(1-C_\alpha)+n_2S_\alpha$	$(n_2(2n_3^2-1)(1-C_\alpha)-n_1n_3S_\alpha)/A$
$A=\sqrt{n_1^2+n_2^2}$ $\quad C_\alpha=\cos\alpha$ $\quad S_\alpha=\sin\alpha$			

$\mathbf{H}$ and the gradient $\nabla(\vec{\omega})$. We use the Levenberg-Marquardt algorithm to solve the overall least-squares minimization problem. In the rest of this chapter, we will refer to the matching technique as the Newton's method with Levenberg-Marquardt minimization, denoted as the N-L-M method.

3.3.2 Heuristics used in Alignment

We use heuristics to address three specific problems. First, because the silhouette of smooth objects has limited information, in order to achieve more accurate pose, we maintain a balance in the importance of silhouette and internal edge alignment as fitting progresses. Second, in order to allow for some occlusion, we provide a scheme for discarding certain edge segments during matching. Third, because our object class does not provide us with salient point features, we use a heuristic that synthesizes "matching points of greatest distance" which are most useful in driving the match error down.

Balancing the Fitting between Internal and Boundary Edges

The Newton's method described above will usually converge as long as the matching contours are the actual corresponding contours. Matching silhouettes is reliable because the boundary edges are well separated [27]. However, for smooth objects, small changes in any rotation parameter may have little effect on the silhouette. Not only are internal edges represented exactly in the model aspect, but they are usually closer to the viewing axis making their images more sensitive to rotation. Including internal edges, when available, in the parameter-fitting process is often necessary for deriving accurate object pose. In doing so, we also need to consider relatively different numbers of internal versus boundary edge segments.

Let B_m and I_m be the boundary and internal edges of the model edgemap, and B_o and I_o the boundary and internal edges of the observed edgemap respectively. Define the mean-squared distances of *certain* boundary edges and internal edges as $D(B_o, B_m(\vec{\omega}))$ and $D(I_o, I_m(\vec{\omega}))$ respectively. Then the merit function in Eq. (3.4) is modified to

$$D(E_o, E_m(\vec{\omega})) = \frac{1}{2}[D(B_o, B_m(\vec{\omega})) + W \cdot D(I_o, I_m(\vec{\omega}))] \tag{3.6}$$

where W is the weight controlling the trade-off between minimizing $D(B_o, B_m(\vec{\omega}))$ and minimizing $D(I_o, I_m(\vec{\omega}))$. For large W, the parameter corrections, $\mathbf{H}^{-1} \cdot [\nabla(\vec{\omega})]$, as in Eq. (3.5), will tend to favor fitting the internal edges; for small W, the parameter corrections will favor minimizing the fitting error in boundary edges. To implement

a balance, we adjust W at each iteration based on the ratio:

$$W^{(k+1)} = \frac{D(I_o, I_m(\vec{\omega}^{(k)}))}{D(B_o, B_m(\vec{\omega}^{(k)}))}. \tag{3.7}$$

This heuristic does not guarantee convergence. However, it ensures that the parameter fitting will not be overly biased to one type of edge contour.

Measuring Errors and Identifying Correspondence

Newton's method would behave well if we had at least three *correct* point correspondences between E_o and E_m from which to compute the error in image matching. Point correspondences are either ill-defined or difficult to find because salient viewpoint-independent features such as line segments and corners may not be available from the edgemaps of a smooth object. Finding point correspondences can be further complicated if there is partial occlusion of the object. This situation suggests that a global correspondence be established rather than local correspondences between features. We first establish global correspondence based on 2D template matching of *internal edges* under the assumption that the matched object and model should have similar contours and the overall distance between the corresponding contours is the smallest of all possible correspondences. This assumption is justified for smooth objects under a viewpoint close to the correct viewpoint. Below, we describe a heuristic that assigns local point correspondences based on the prior establishment of a good global correspondence between the predicted edgemap E_m and the observed edgemaps E_o. These synthesized local correspondences are then used to iteratively drive down the

fit error.

Let I_o and I_m be the internal edges of the observed edgemap and the model edgemap respectively (see Figure 3.1(a) and (b)). Matching starts by sliding I_m over I_o in 2-space searching for a position where the total squared error distance of the corresponding contours reaches its minimum. The error distance of the edge point in I_m to the nearest point in I_o can be computed in constant time by using a chamfering technique (this technique is also used by Huttenlocher *et al.* [59] in the computation of Hausdorff distance). The chamfering technique [7] expands each object contour point in all directions with an increasing value of gray level which represents the distance from the observed object contours. The distance function yields the distance, measured in pixels, from any image pixel to the nearest contour point. After chamfering the observed contours once, the match error for N model points can be computed in $O(N)$ time.

Once the global correspondence is established as above, we divide the bounding box of the model edgemap into 36 regions (could be k by k for another k) and locate contour segments in each region if they exist (Figure 3.1(c)). On each model segment, we designate the point that has the *largest* error distance as the search point, and then use this point and the obtained error distance to search for a corresponding point on the observed edgemap (Figure 3.1(d)). In a situation when the observed object is partially occluded, the edgemap of the occluded portion does not have matches in the model edgemap and tends to have large error distances in template matching. To avoid including these incorrect point correspondences in Newton's method, we select as matches the best 50% of the point correspondences on the silhouettes and

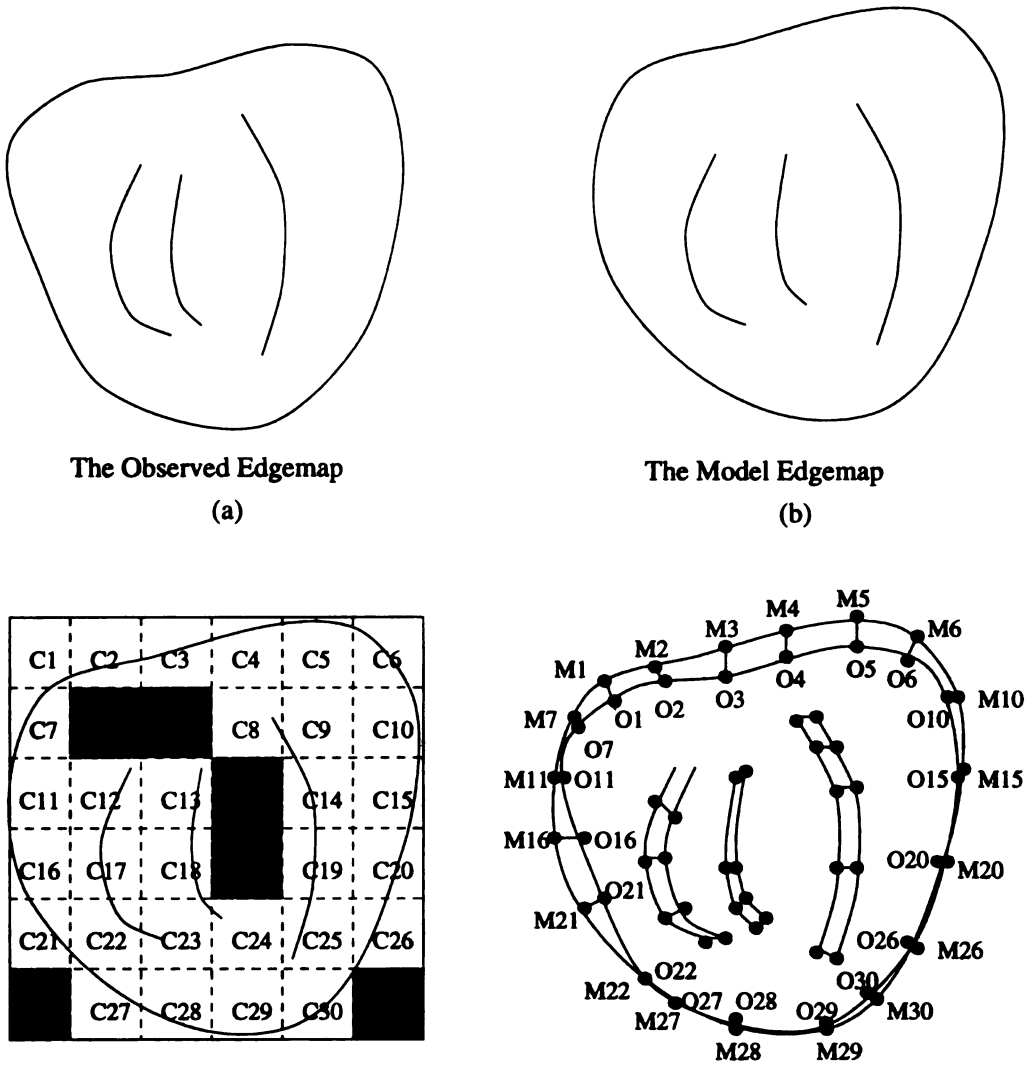


Figure 3.1: Synthesizing local point correspondences. (a) The observed edgemap E_o. (b) The model edgemap E_m. (c) The bounding box of E_m is divided into 36 regions; the region is labeled if it contains a contour segment. (d) The selected point correspondence in each region if applicable (only silhouette point correspondences are labeled).

the internal edges respectively. The error distance on each selected contour segment will force the pose parameters in Newton's method to be adjusted in such a way that at each iteration the model contour segments will get closer to the observed contour segments. The matching algorithm is sketched in Figure 3.2.

Note that the template matching would fail to synthesize local point correspondences if the observed edgemap and the model edgemap were not in an approximate scale. Here, we assume that the initial scale of the observed object is given. Determination of the initial scale value for matching will be discussed in Chapter 4. As an example of showing that the scale is not fixed, consider the car (Ford Taurus) in Figure 3.3(a). Figure 3.3(b) shows the model edgemap of the car which is 2 times larger than the observed edgemap. The initial parameter value for scale was set to 0.5. Figure 3.3(c) shows the evolution of the model edgemaps generated during the iterations of matching. The white contour indicates the final edgemap generated and is shown superimposed on the original intensity image of the car as in Figure 3.3(d).

3.3.3 The Sensitivity of Pose Parameters to Object Shape

We show that the sensitivity of the rotation parameters is closely related to the object shape. As mentioned earlier, silhouettes do not provide enough information for accurate pose estimation, and including internal edges in the matching process is often a necessity. Directly below we provide a theoretical characterization and later on we show experimental evidence for this.

The basic idea of the sensitivity analysis for smooth objects is shown in Figure 3.4.

Input: an observed edgemap E_o, a model edgemap E_m,
an initial estimate of pose parameters $\vec{\omega} = (t_x, t_y, s, \theta, \phi, \alpha)$,
an image matching error threshold E_t in pixel units, and
a limit on the number of iterations I_{max} for failure.

Output: either failure or the best-fit pose parameters $\vec{\omega}$.

Processing: this algorithm computes $\vec{\omega}$ iteratively using Newton's method until the mean-squared distance $D(E_o, E_m(\vec{\omega}))$ is below E_t or until the maximum number of iterations I_{max} have been tried.
In the first case, we have alignment, and in the second case failure.
The iterations repeat the following steps:

(1) Establish global correspondence based on the template matching of internal edges (or boundary edges in the case when only silhouettes are used in alignment);

(2) Divide the bounding box of E_m into 36 regions;

(3) In each region applicable, locate the point correspondence that has the largest error distance (point correspondences are assigned separately for internal edges and silhouettes);

(4) Select the best 50% of the point correspondences on the silhouettes and the internal edges respectively;

(5) Use the point correspondences found in steps 3 and 4 to compute the weight W in Eq. (3.7) and also to derive the Hessian matrix $\mathbf{H}$ and the gradient $\nabla(\vec{\omega})$, and then recompute $\vec{\omega}$ using Eq. (3.5);

6) Use $\vec{\omega}$ to predict the new E_m and compute the modified $D(E_o, E_m(\vec{\omega}))$ as in Eq. (3.6) on those contour segments in selected regions.

Figure 3.2: The edgemap alignment algorithm.

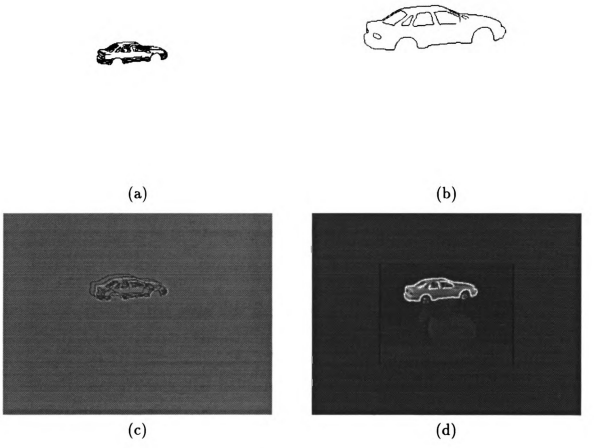


Figure 3.3: Matching edgemaps in an 1:2 scale. (a) The observed edgemap. (b) The model edgemap. (c) The evolution of convergence in the matching algorithm. (d) The fitted edgemap shown superimposed on the original image.

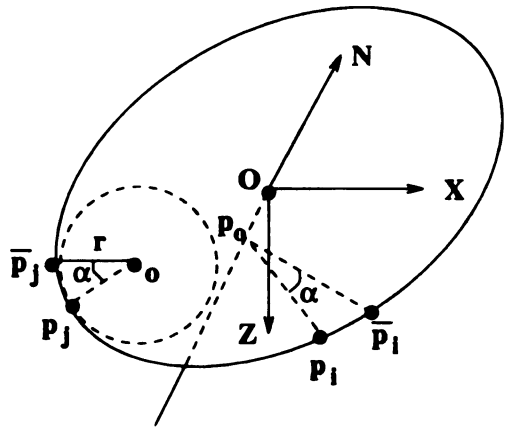


Figure 3.4: The rotation of a smooth object.

Let X and Y be the main axes of the image plane, and let the Z axis be the visual axis. Consider an object rotated through some α angle about a rotation axis $\vec{\mathbf{N}}$ with direction $(n_1, n_2, n_3)^t$. Let $\mathbf{p_j}$ be a rim point and $\bar{\mathbf{p}}_\mathbf{j}$ be the corresponding rim point after rotation. Let $\mathbf{p_i}$ be an internal edge point and $\bar{\mathbf{p}}_\mathbf{i}$ be the corresponding point after rotation. Let $\mathbf{r}$ be the curvature radius at $\bar{\mathbf{p}}_\mathbf{j}$. Note that $\mathbf{r}$ is a vector perpendicular to both $\vec{\mathbf{N}}$ and Z. The point $\mathbf{o}$ is the center of the circle of curvature of $\mathbf{p_j}$. Let $\Delta\mathbf{p_j} = (\Delta x_j, \Delta y_j, \Delta z_j)^t$ denote $\bar{\mathbf{p}}_\mathbf{j} - \mathbf{p_j}$ and $\Delta\mathbf{p_i} = (\Delta x_i, \Delta y_i, \Delta z_i)^t$ denote $\bar{\mathbf{p}}_\mathbf{i} - \mathbf{p_i}$. Without loss of generality, let us assume that the rotation axis passes through the origin $\mathbf{O}$ and points $\mathbf{p_i}$, $\bar{\mathbf{p}}_\mathbf{i}$, $\mathbf{p_j}$, and $\bar{\mathbf{p}}_\mathbf{j}$ are on the same cross section. Let $\mathbf{p_o}$ be the rotation center of the cross section through points $\mathbf{p_i}$, $\bar{\mathbf{p}}_\mathbf{i}$, $\mathbf{p_j}$, and $\bar{\mathbf{p}}_\mathbf{j}$. The following statements hold for rotation $\mathbf{R}$.

(1) $\Delta\mathbf{p_j}$ lies in a plane perpendicular to the rotation axis. That is, $\Delta\mathbf{p_j} \cdot \vec{\mathbf{N}} = 0$.

(2) Points lying on the rotation axis $\vec{\mathbf{N}}$ are invariant to rotation $\mathbf{R}$. That is, $\mathbf{R}(\lambda\vec{\mathbf{N}}) = \lambda\vec{\mathbf{N}}$ or $(\mathbf{R} - \mathbf{I})(\lambda\vec{\mathbf{N}}) = \mathbf{0}$ where λ is any real number, and $\mathbf{I}$ is the

3×3 identity matrix.

Note that points $\mathbf{p_j}$, $\bar{\mathbf{p}}_\mathbf{j}$, and $\mathbf{o}$ form a plane that is perpendicular to both $\vec{\mathbf{N}}$ and the Z axis, and a *cross section* is defined as the intersection of the plane with the object.

Proposition 3.1 *If point* $\mathbf{o}$ *lies on the rotation axis* $\vec{\mathbf{N}}$, *then* $\mathbf{p_j}$ *is invariant to the pose parameter* α.

Proof: From the curvature method in Eq.(2.2), we have

$$\bar{\mathbf{p}}_\mathbf{j} = \mathbf{R}(\mathbf{p_j} - \mathbf{r}) + \mathbf{r}.$$

Rewriting the above equation, we obtain

$$\bar{\mathbf{p}}_\mathbf{j} - \mathbf{p_j} = (\mathbf{R} - \mathbf{I})(\mathbf{p_j} - \mathbf{r})$$

Since $\mathbf{o}$ lies on the rotation axis $\vec{\mathbf{N}}$, we have $\mathbf{p_j} - \mathbf{r} = \lambda\vec{\mathbf{N}}$ for some real number λ. From Fact (2), we obtain $(\mathbf{R} - \mathbf{I})(\mathbf{p_j} - \mathbf{r}) = \mathbf{0}$, i.e., $\Delta\mathbf{p_j} = 0$. Thus, $\mathbf{p_j}$ is invariant to the pose parameter α. □

From Proposition 3.1, we can see that the image of point $\mathbf{p_j}$ will be of no help in refining α in the optimization because it does not change with α when rotated. Equivalently, $\frac{\Delta\mathbf{p_j}}{\Delta\alpha}$ is unaffected by the rotation. Thus, for a symmetric object such as a circular cone, it is obvious that rotating the cone around its principle axis results in no rotation angle information. Proposition 3.1 is clearly true for a totally symmetric

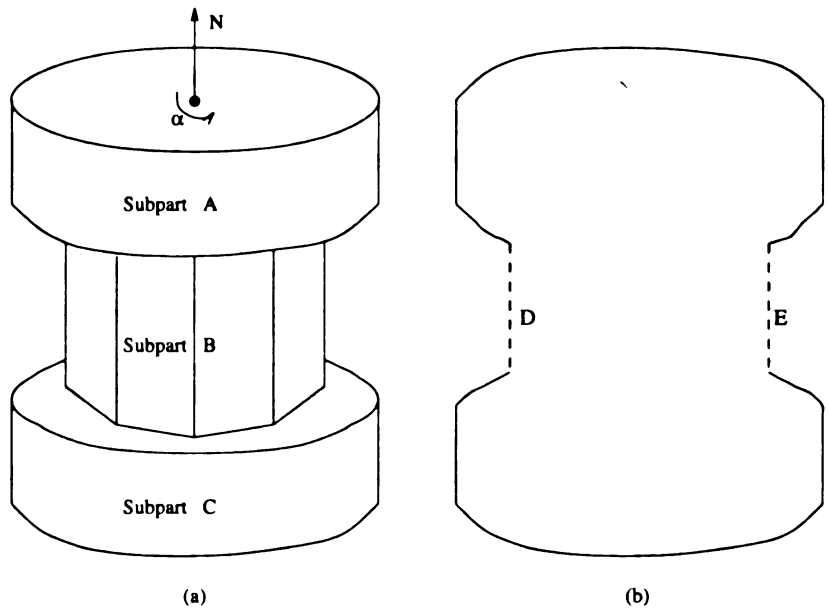


Figure 3.5: A scene where the matching algorithm fails to recover the pose parameter α. (a) The object is composed of three subparts. (b) The extracted silhouette of the object.

object, such as a sphere, rotating about *any* axis: there is no change of silhouette shape with respect to any rotation angle α. In fact, this proposition is consistent with the prediction error of the curvature method in Eq. (2.9), where $E(\frac{c^2}{a^2}, \alpha) = 0$ when $a = c$, i.e., the cross section is a circle, which makes α irrelevant (and thus insensitive) to the predicted appearance.

Definition 3.1 *The local curve, from* $\mathbf{p_j}$ *to* $\bar{\mathbf{p}}_\mathbf{j}$, *is said to be invariant to rotation* $\mathbf{R}$ *up to a rotation angle of* α *if the center of the circle of curvature* $\mathbf{o}$ *lies on the rotation axis.*

If many local curves of an observed object are invariant to rotation $\mathbf{R}$ up to some rotation angle, then the pose parameter α can not be recovered accurately in the parameter fitting process. As an example, consider the object in Figure 3.5. Subparts A and C are circular cylinders with their principle axes coincident with the rotation

axis. Subpart B is a cylinder with octagonal cross section. It is obvious that the silhouettes of subparts A and C are invariant to rotation angle α, while the silhouette of subpart B varies only within a rotation angle of 45 degrees. For designating point correspondences on silhouettes, the contour segmentation scheme, as described in the matching algorithm in Figure 3.2, will choose the best 50% contour segments, in which the error measurements are among the smallest. In this case, contour segments D and E, as in Figure 3.5(b), will not be selected (they will be treated as occluded portions of the object because of their large error measurements). Then the silhouettes selected for matching will remain unchanged regardless of the rotation angle α, resulting in a loss of rotation angle information. Thus, our matching heuristics for the silhouette behave badly in this case. However, α is sensitive to the internal edges within the 45° ambiguity intervals.

Proposition 3.2 *Let* $\mathbf{p_i}$ *be the internal edge points such that* $\mathbf{p_i} \neq \lambda\vec{\mathbf{N}}$ *for some* $\lambda \neq 0$. *Then the pose parameter* α *is sensitive to* $\mathbf{p_i}$.

Proof: Since $\mathbf{p_i}$ is an internal edge point, the radius of curvature $\mathbf{r} = \mathbf{0}$. Then the curvature method becomes

$$\bar{\mathbf{p}}_\mathbf{i} = \mathbf{R}\mathbf{p_i}$$

for internal edge point $\mathbf{p_i}$. Rewriting the above equation, we obtain

$$\Delta\mathbf{p_i} = \bar{\mathbf{p}}_\mathbf{i} - \mathbf{p_i} = (\mathbf{R} - \mathbf{I})\mathbf{p_i}.$$

Since $\mathbf{p_i} \neq \lambda\vec{\mathbf{N}}$ for some $\lambda \neq 0$, thus $\|\mathbf{p_i} - \mathbf{O}\| > 0$ (i.e., $\mathbf{p_i} \neq \mathbf{O}$). It follows

immediately that $\Delta \mathbf{p_i} \neq \mathbf{0}$. Thus, the pose parameter α is sensitive to $\mathbf{p_i}$. □

Proposition 3.2 also suggests that the pose parameter α is insensitive to any internal edge point lying on the rotation axis. But they occur only at the intersections of the rotation axis and the object surfaces because internal edge points are surface points. For smooth objects, there are usually at most two such points existing (there would be many for a polyhedron rotating about an edge). Moreover, our heuristics in designating point correspondence on internal edge points will exclude these points for alignment. Thus, the pose parameter α can be recovered for somewhat symmetric and smooth objects if internal edges are used in the matching process. Accuracy depends on the number of such points $\mathbf{p_i}$ selected and the size of $\mathbf{\Delta p_i}$ relative to the pixel size.

Proposition 3.3 *Let* $\mathbf{p} = (x, y, z)^t$ *and* $\bar{\mathbf{p}} = (\bar{x}, \bar{y}, \bar{z})^t$ *be the corresponding points before and after the rotation. Let the rotation axis be the Y axis, and the rotation angle be* α. *Let* $\mathbf{p_o} = (0, y_o, 0)^t$ *be the rotation center of the horizontal cross section through* $\mathbf{p}$. *Let* $\mathbf{\Delta p} = (\Delta x, \Delta y, \Delta z)$ *denote* $\bar{\mathbf{p}} - \mathbf{p}$. *Let* l *be the length of the vector* $\vec{\mathbf{p_o p}}$ *and* β *be the angle between the X axis and the vector* $\vec{\mathbf{p_o p}}$. *Then*

$$\Delta x = -2l \sin(\frac{\alpha}{2} + \beta) \sin \frac{\alpha}{2}.$$

Proof: Note that the points $\mathbf{p}$, $\bar{\mathbf{p}}$, and $\mathbf{p_o}$ all lie on the same horizontal section implying that $y = \bar{y} = y_o$. Since the object is rotated through angle α about the

Y axis, we only need to consider the relationship between the coordinates (x, z) of the original point and $(\bar{x}, \bar{z})$ of the rotated point. Using polar coordinates in the cross section, a point with polar coordinates (l, β) rotates to $(l, \beta + \alpha)$ where $l = \sqrt{x^2 + z^2}$ and $\beta = \tan^{-1}(z/x)$. Now $x = l\ \cos\beta$, so that $\bar{x} = l\ \cos(\beta + \alpha)$. Hence $\Delta x = \bar{x} - x = -2l \sin(\frac{\alpha}{2} + \beta) \sin\frac{\alpha}{2}$. □

It can be inferred from Proposition 3.3 that for a fixed l and a small non-zero α, $|\Delta x|$ is close to its maximum when β is close to $\frac{\pi}{2}$ or $\frac{3\pi}{2}$, and $|\Delta x|$ is close to 0 when β is close to 0 or π. By the definition of the rim (i.e., the set of all points on the visual surface, whose normal is perpendicular to the visual axis), β is either 0 or π for boundary points. Thus, the rotation is less sensitive to points on the boundary than to any other points lying on the same cross section. In fact, for a symmetric object, we learn from Propositions 3.1 and 3.2 that when the object is rotated, the boundary points remain unchanged, but not the internal surface points.

For simplicity of the analysis, let us assume that the object is rotated through α about the Y axis. All the notations used here are consistent with the ones in Figure 3.4. Let l_i and l_j denote the length from $\mathbf{p_o}$ to $\mathbf{p_i}$ and $\mathbf{o}$ respectively, i.e., $l_i = \|\vec{\mathbf{p_o p_i}}\|$ and $l_j = \|\vec{\mathbf{p_o o}}\|$. Let β_j be the angle between the X axis and the vector $\vec{\mathbf{p_o o}}$, and β_i the angle between the X axis and the vector $\vec{\mathbf{p_o p_i}}$. From Proposition 3.3, we have $\Delta x_i = -2l_i \sin(\frac{\alpha}{2} + \beta_i) \sin\frac{\alpha}{2}$ and $\Delta x_j = -2l_j \sin(\frac{\alpha}{2} + \beta_j) \sin\frac{\alpha}{2}$. Then, $|\Delta x_j| \leq |\Delta x_i|$ if $|l_j \sin(\frac{\alpha}{2} + \beta_j)| \leq |l_i \sin(\frac{\alpha}{2} + \beta_i)|$. Note that for rim point $\mathbf{p_j}$, the center of the circle of curvature $\mathbf{o}$ is used to compute Δx_j because $\mathbf{r}$ remains unchanged after rotation. For a very smooth object, if the object's center of gravity is assumed to be the rotation center, then both l_j and β_j tend to be small but $\mathbf{r}$ tends to be large,

causing $|\Delta x_j|$, the projection of the changes in the image, to be small. On the other hand, l_i does not depend on the local curvature at $\mathbf{p_i}$, thus Δx_i depends on the values of β_i and l_i. In most cases, for this kind of object, $l_i > l_j$ and $|\beta_i| > |\beta_j|$, and consequently, $|\Delta x_i| \geq |\Delta x_j|$. Thus, the pose parameter α is more sensitive to the internal edge points than to the silhouette points. An interesting case to consider is a cube rotating about an axis through the center of two opposite faces. All silhouette points have $\mathbf{r} = \mathbf{0}$. For some aspects, α is more sensitive to silhouette points, and for other aspects, α is more sensitive to internal edge points.

3.3.4 The Basin of Convergence

In this section, we assume that the ground truth of pose parameter estimates is known, and we investigate the basin of convergence. In our implementation of nonlinear least squares minimization, we use the Levenberg-Marquardt method to search for a minimum. The Levenberg-Marquardt method is one of the most widely used methods for nonlinear least squares minimization. This method uses a scalar parameter λ to force iterative convergence. The new form of the Hessian matrix $\mathbf{H}$ in Eq. (3.5) is given by

$$H'_{kl} = \begin{cases} (1+\lambda)H_{kl} & \text{when } k = l \\ H_{kl} & \text{when } k \neq l \end{cases} \tag{3.8}$$

where $H_{kl}, k, l = 1, \cdots, 6$, are the array elements of $\mathbf{H}$. As λ is increased, the solution increasingly corresponds to pure gradient descent. For decreasing λ, the problem would gradually move back to the original Newton's method. This phenomenon can

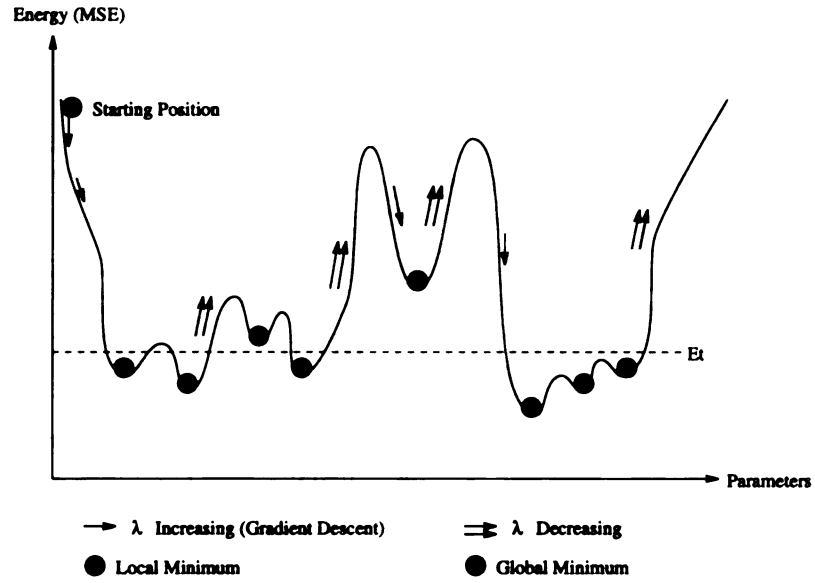


Figure 3.6: The basin of convergence.

be described as in Figure 3.6. A ball is moving downward on a hill by increasing λ until it reaches the valley, and then starts moving upward on a hill by decreasing λ. If the force of moving upward is not sufficient to allow the ball to reach the top of the hill, the ball will get stuck in the bottom of the valley. If the force is powerful enough, the solution is searched for in the next valley. This will continue as long as the uphill force is large enough. If the uphill of a global minimum is shallow, the search may not stop at the global minimum. Thus, the iterations should be terminated when the mean-squared distance error (MSE) is smaller than some error threshold Et.

Those valleys where the parameter estimates satisfy some error criteria in *parameter space* are defined as *global minima*, otherwise, they are defined as *local minima*. For example, suppose that $\hat{\omega}$ and $\bar{\omega}$ are the pose parameter estimates and the ground truth of pose parameters, respectively; let Eb be the error bounding box in parameter space; then global minima are defined as those pose parameter estimates $\hat{\omega}$ satisfying

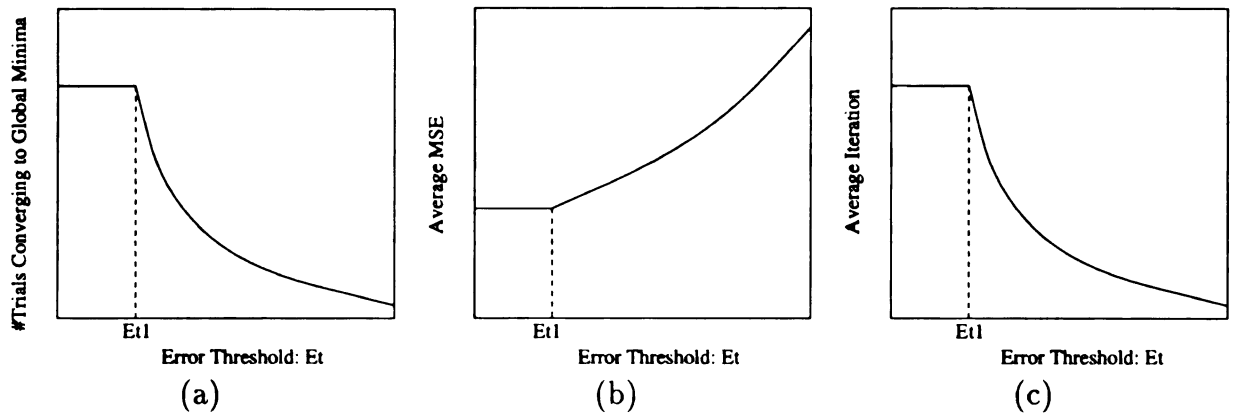


Figure 3.7: The effect of Et on the basin of convergence. (a) The number of trials converging to global minima. (b) The average MSE error of fitting. (c) The average converging time of fitting.

$|\hat{\omega} - \bar{\omega}| \in Eb$. We implicitly assume that the fitting errors (MSE) of global minima are smaller than those of local minima; this assumption is justified for pose parameter estimates close to the ground truth which make the fitting error small.

A valley with MSE below Et becomes an attractor, absorbing any ball in the valley. Consider the number of searches which would terminate in global minima if started at random in some region of parameter space (or equivalently, consider the *probability* of searches reaching global minima). When Et is below the MSE of the *actual global minimum* (i.e., the ground truth of pose parameters), denoted as $Et1$ (see Figure 3.7(a)), the number of global minima achieved remains fixed because there exist no attractors and converging to global minima depends solely on the topology of the basin and the initial starting positions. When Et is increased, so is the number of attractors which may include some valleys of local minima, and thus, the number of searches converging to global minima is decreased. This phenomenon also explains why the average MSE of convergence is increased as Et is increased (see

Figure 3.7(b)). The phenomenon depicted in Figure 3.7(c) is obvious because more and more attractors exist as Et is increased further, and consequently, the average number of iterations to convergence is decreased. Our results below illustrate these concepts and also show how Et might be chosen in practice.

3.4 Experimental Results

The alignment algorithm presented in this chapter has been tested for both real sculptured objects and synthetic objects made from a superquadric model [5]. The algorithm has succeeded in deriving the object pose by aligning only the silhouettes [27]. In this section, we intend to demonstrate: (1) the algorithm can handle partially occluded objects and objects with internal edges; (2) the basin of the convergence is broad; (3) the accuracy of the derived object pose can be further improved by introducing object internal edges in the alignment process.

As an example of showing the intermediate stages of the alignment process, consider the superquadric model, an *ellipsoid*, in Figure 3.8(a). Figure 3.8(b) shows the extracted edgemap of the ellipsoid. Figure 3.8(c) shows the evolution of the model edgemaps generated during the iterations of N-L-M method. The white contour indicates the final edgemap generated from N-L-M method. Figure 3.8(d) shows this edgemap superimposed on the observed object. Figure 3.9 gives another example of this alignment process. As can be seen in Figure 3.9(b), the observed object, a partially occluded *block*, is fitted by the model within only a few iterations as indicated by the evolution of model edgemaps in Figure 3.9(a).

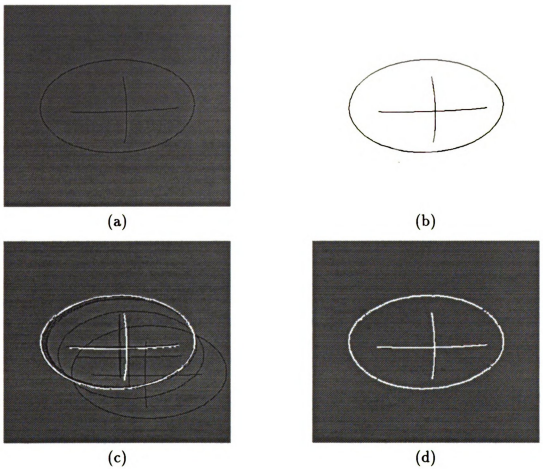
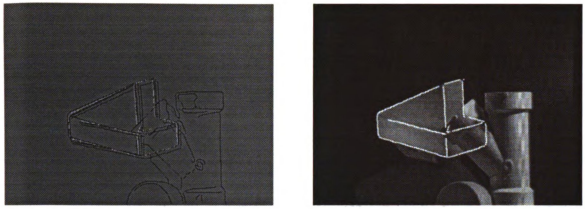


Figure 3.8: Steps in the model fitting algorithm. (a) Intensity image synthesized from a superquadric model. (b) Edgemap extracted from the image of (a). (c) The evolution of convergence in the N-L-M method. (d) The fitted edgemap shown superimposed on the original image.



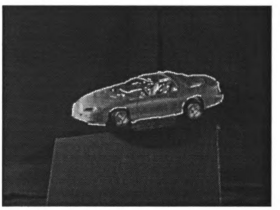
(a) (b)

Figure 3.9: Fitting a polyhedral model to a partially occluded object. (a) The evolution of convergence in the N-L-M method. (b) The fitted edgemap shown superimposed on the original image. (Note the occlusion by the Y pipe.)

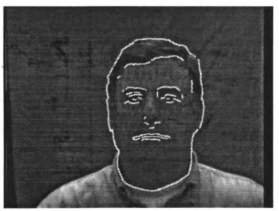
3.4.1 Results from Model Fitting Experiments

The alignment algorithm for locating objects in the scene was tested on a total of sixty aspect models. Half of them are partially occluded objects. Figures 3.10 and 3.11 show a few of alignment examples. Note that in several of these images, the test objects are partially occluded. As can be seen from the white contours in Figures 3.10 and 3.11, most of the observed objects are well fitted. Table 3.3 shows the goodness of the fit. Note that $E_{\mathbf{N}}$ is defined the angle between $\mathbf{N}$ and $\bar{\mathbf{N}}$, i.e., $\cos^{-1} \frac{\mathbf{N} \cdot \bar{\mathbf{N}}}{\|\mathbf{N}\| \|\bar{\mathbf{N}}\|}$, where $\bar{\mathbf{N}}$ as the ground truth for $\mathbf{N}$. The translational parameters, (t_x, t_y), are recovered within 3 pixels, the rotation angle, α, within 3 degrees, the rotation axis, $\mathbf{N}$, within 0.2 radians and the scaling factor within 0.01. The final 2 columns of Table 3.3 show that alignment is achieved within 1.6 pixels MSE on the edgemaps within 10 iterations of the hill-climbing procedure.

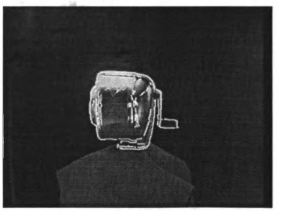
(a) can



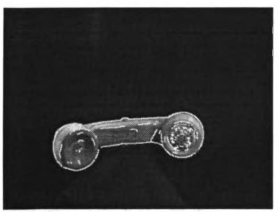
(b) camaro8



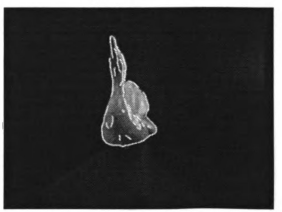
(c) face



(d) sharpener10

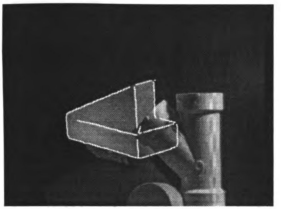


(e) phone6



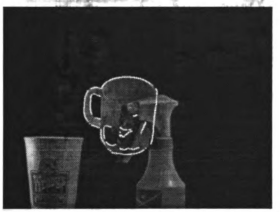
(f) swan11

Figure 3.10: Model fitting examples of sculptured objects.



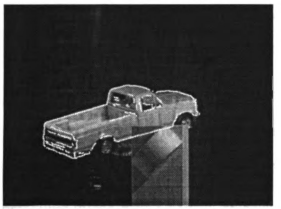
(a) blockA1

(b) lion29



(c) mug26

(d) pig1



(e) truck10

(f) zebra1

Figure 3.11: Model fitting examples of occluded objects.

Table 3.3: Results from Model Fitting.

Input parameters: Et = 1.6 pixels; Imax = 30 iterations							
Object	Pose Estimation Error					Number of Iterations	MSE
	E_{t_x} (pixel)	E_{t_y} (pixel)	E_s	E_α (degree)	$E_{\mathbf{N}}$ (radian)		
can	0	0	0.003	0.82	0.05	4	0.94
camaro8	3	2	0.009	0.37	0.06	9	1.16
face*	0	0	0.002	2.20	0.20	7	1.39
phone6	1	0	0.001	0.40	0.06	6	1.08
sharpener10	3	0	0.001	2.39	0.13	8	1.36
swan11	0	0	0.001	1.05	0.15	4	0.99
blockA1	0	2	0.007	0.38	0.16	5	1.02
lion29	1	0	0.001	0.49	0.06	5	1.56
mug26	2	0	0.010	0.18	0.10	7	0.98
pig1	0	0	0.001	0.07	0.03	5	1.22
truck10	2	2	0.009	0.74	0.07	10	0.95
zebra1	3	3	0.010	0.31	0.05	8	0.76
$E_p = \|\hat{p} - p_{true}\|$ where $\hat{p}$ is the estimated parameter value of p_{true} *ground truth not actually known (only approximation).							

3.4.2 Results from Monte Carlo Experiments

To investigate the convergence characteristics within a viewing aspect, the parameters of rotation, α, θ, and ϕ, in the aspect were sampled at equal intervals covering the whole viewing aspect. Each sample of these three parameters, together with the translation parameters (t_x, t_y) and the scaling parameter s, forms the initial parameter estimates for the fitting. The translational parameters (t_x, t_y) were randomly selected within $[-20, 20]$ pixels and s within $[0.9, 1.1]$. Note that the indexing scheme described in the next Chapter will justify using such initial estimates. 240 samples were generated for each viewing aspect. Six test objects, as shown fitted by their corresponding models in Figure 3.12, were used to conduct the experiments. Three of them are symmetric along some rotation directions. The purpose for selecting such

Table 3.4: Convergence and Error Analysis: Edgemap vs. Silhouette.

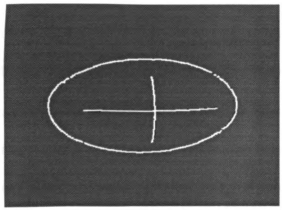
Number of trials: 240 Et = 1.0 pixels; Imax = 30 iterations; Eb is as defined in Eq. (3.9)			
	Silhouette and Internal Edges		
Object	Avg Iteration	♯ (Error < Et)	$\sharp\lvert\omega - \bar{\omega}\rvert \in Eb$
ellipsoid	9	240	240
cup	12	240	210
taurus1	10	240	182
phone13 (occluded)	12	240	195
squash (occluded)	9	240	232
squirrel (occluded)	11	240	173
	Silhouette Only		
Object	Avg Iteration	♯ (Error < Et)	$\sharp\lvert\omega - \bar{\omega}\rvert \in Eb$
ellipsoid	9	240	17
cup	6	240	44
taurus1	9	240	134
phone13 (occluded)	11	240	145
squash (occluded)	10	240	220
squirrel (occluded)	16	240	143

objects is to study the effect of object shape on the accuracy of the estimated object pose.

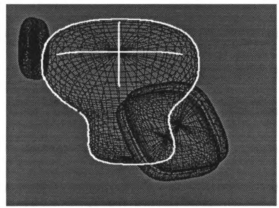
Table 3.4 lists the number of trials converging: $\sharp(Error < Et)$, and the number of the trials converging to global minima: $\sharp|\omega-\bar{\omega}| \in Eb$ where ω and $\bar{\omega}$ are the estimated pose parameters and the ground truth parameters, respectively, and Eb is the error bounding box in parameter space centered around the ground truth parameters $\bar{\omega}$. The criteria for global minima are, i.e., the range of Eb:

$$\begin{aligned} &|t_x - \bar{t_x}| \leq 3 \text{ (pixels)}, \quad |t_y - \bar{t_y}| \leq 3 \text{ (pixels)}, \qquad |s - \bar{s}| \leq 0.01, \\ &|\alpha - \bar{\alpha}| \leq 3 \text{ (degrees)}, \quad \|\mathbf{N} - \bar{\mathbf{N}}\| \leq 0.2 \text{ (radians)}, \end{aligned} \tag{3.9}$$

where $\mathbf{N}(\theta, \phi) = (\cos\theta \sin\phi, \sin\theta \sin\phi, \cos\phi)$ is the direction of the rotation axis, and



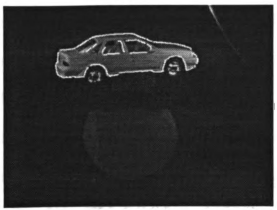
(a) ellipsoid



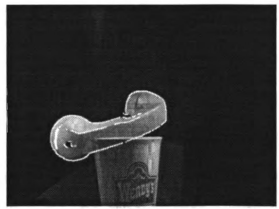
(b) squash

(c) cup

(d) squirrel



(e) taurus1



(f) phone13

Figure 3.12: Test objects used for studying the basin of convergence.

$\bar{\alpha}$, $\bar{\mathbf{N}}$, $\bar{s}$, $\bar{t_x}$, and $\bar{t_y}$ the ground truth parameters. $\|\mathbf{N} - \bar{\mathbf{N}}\|$ denotes the angle between $\mathbf{N}$ and $\bar{\mathbf{N}}$, i.e., $\cos^{-1} \frac{\mathbf{N} \cdot \bar{\mathbf{N}}}{\|\mathbf{N}\| \|\bar{\mathbf{N}}\|}$. As can be seen in the column $\sharp(Error < Et)$ of Table 3.4, all 240 trials in each object experiment converged to (either global or local) minima within an error threshold of 1.0 pixels, suggesting a broad basin of convergence for the listed 6 object aspects. The quantity, $\sharp|\omega - \bar{\omega}| \in Eb$, indicates the number of converged trials that actually converge to global minima. This quantity also indicates the sensitivity of pose parameters to object shape. Consider the ellipsoid in Figure 3.12(a) for an example. As shown in Table 3.4, all trials in the ellipsoid experiment converged to global minima when internal edges together with silhouette were used in the fitting process. This is because the ellipsoid is made from a superquadric model and the object can be precisely modeled. However, when the silhouette alone was used in the fitting process, only 17 (out of 240) trials converge to global minima. The deterioration of converging to global minima is attributed to the symmetry of the ellipsoid and the insensitivity of rotation parameters to the object silhouette. This phenomenon is also depicted in Figure 3.8(c) where the model silhouette converges to the object silhouette after 3 iterations but the pose is still far from the ground truth as indicated by the cross mark. Another experiment showing this phenomenon is conducted via the cup in Figure 3.12(c). The object is almost symmetric around the rotation axis, and as a consequence, only 44 (out of 240) trials converged to global minima when only the silhouette was used in the fitting process. But the situation is much improved when internal edges were incorporated into the fitting process as shown in Table 3.4. 210 (out of 240) trials were able to converge to global minima. The squash in Figure 3.12(b) is a very smooth and symmetric object

made from a superquadric model. Although occluded, 232 (out of 240) trials are able to converge to global minima when both silhouette and internal edges were used in the fitting process. The lack of internal edges in the fitting process does not decrease the number of convergence to global minima significantly. This is because that, unlike the ellipsoid and the cup, the squash is not symmetric about the rotation axis and also the occlusion helps to break the symmetry of the shape.

It is observed in Section 3.3.3 that when objects are rotated, their silhouettes may vary little while their internal edges may move significantly in the image. Thus, including the internal edges in the fitting process should be helpful in determining the object orientation. We have demonstrated in the above three experiments that the pose parameters obtained by fitting internal and boundary edges simultaneously are much more accurate than by fitting the boundary edges alone. Experiments conducted on the other three objects, *squirrel, taurus1* and *phone13* as in Figures 3.12(d), (e) and (f), respectively, also reflect this property that incorporating internal edges in the fitting process can help improve the accuracy of pose parameters (see Table 3.4). Since these three objects are not symmetric about the rotation axis, the convergence rate to global minima does not deteriorate as much as that for the symmetric objects.

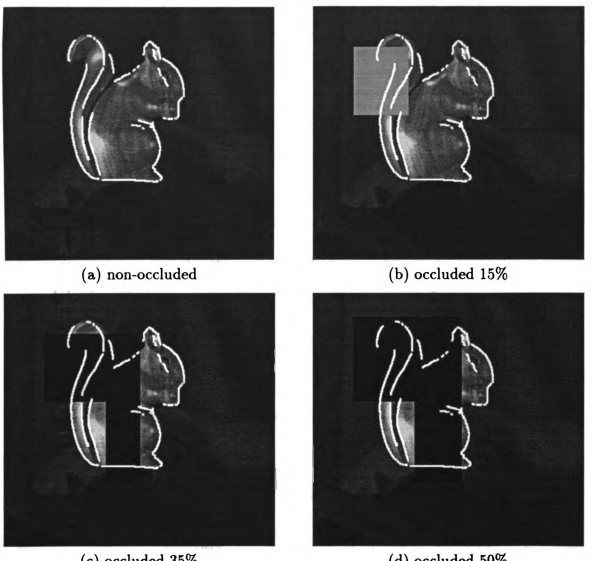
Table 3.5 lists the number of trials converging (N_{conv}), the average iteration (I_{avg}), the average MSE (E_{avg}), and the number of trials converging to global minima ($\sharp|\omega - \bar{\omega}| \in Eb$). We can make the following observation from Table 3.5: (i) all trials converge regardless of the value of Et, indicating the basin of convergence is broad. (ii) The number of average iterations decreases as Et becomes more tolerant. This attributes to the fact that when the MSE error threshold Et becomes more tolerant,

Table 3.5: Convergence and Pose Accuracy vs. Et.

Test object: taurus1 (in Figure 3.12(e)). Number of trials: 240 Imax: 30 iterations; Eb is as defined in Eq. (3.9)								
Et	Silhouette and Internal Edges				Silhouette Only			
	N_{conv}	I_{avg}	E_{avg}	$\sharp\lvert\omega - \bar{\omega}\rvert \in Eb$	N_{conv}	I_{avg}	E_{avg}	$\sharp\lvert\omega - \bar{\omega}\rvert \in Eb$
1.5	240	5.5	1.3	33	240	4.4	1.3	29
1.4	240	6.4	1.2	52	240	5.2	1.2	33
1.3	240	7.4	1.2	86	240	6.2	1.2	44
1.2	240	8.3	1.1	113	240	7.3	1.1	60
1.1	240	9.6	1.0	145	240	8.7	1.0	92
1.0	240	10.8	0.9	182	240	9.3	0.9	134

more and more local minima attractors become effective, and the fitting process is terminated once it gets stuck in these local minima. This fact also contributes to the following two observations: (iii) the average MSE increases as Et increases; and (iv) the number of trials converging to global minima declines as Et is more tolerant. All these results are consistent with the theoretical discussion given in Section 3.3.4.

As an example to see the effect of occlusion on convergence, we manually occluded a *squirrel* by 0%, 15%, 35%, and 50% respectively as seen in Figure 3.13. Table 3.6 shows that for these cases all 240 trials converged, indicating the basin of convergence is still broad even under severe object occlusion. Table 3.6 also shows that for these cases the number of trials converging to global minima decreases as the percentage of the occlusion increases. This is due to the limited information provided by the unoccluded portion of the object; less constraints enforced on the fitting lead to more local minima attractors which cause fast convergence but deteriorate the performance on the number of trials converging to global minima.



(a) non-occluded

(b) occluded 15%

(c) occluded 35%

(d) occluded 50%

Figure 3.13: Objects for studying the effect of occlusion.

Table 3.6: Convergence and Error Analysis: Occlusion Effect.

Test object: squirrel (in Figure 3.13) Number of trials: 240 Et = 1.0 pixels; Imax = 30 iterations; Eb is as defined in Eq. (3.9)			
Occlusion Percentage(%)	Silhouette and Internal Edges		
	Avg Iteration	$\sharp$ (Error < Et)	$\sharp\lvert\omega - \bar{\omega}\rvert \in Eb$
0	14	240	234
15	12	240	228
35	11	240	173
50	9	240	108

3.5 Summary

In this chapter, a method for locating 3D objects with arbitrary curved surfaces from a single 2D intensity image is outlined. Given an input image and a candidate object model and aspect, the method will verify whether or not the object is present and if so, report pose parameters. The modeling technique of Basri and Ullman [9], as described in Chapter 2 and the alignment method of Lowe [77] are combined together to produce a new technique for handling curved 3D objects. The model allows an object edgemap to be predicted from pose parameters. Pose is computed via an iterative search for the best pose parameters. Heuristics are used so that alignment can succeed in the presence of occlusion and artifact and without resorting to use of corresponding salient feature points.

Alignment experiments with both real and synthetic objects demonstrate a high rate of convergence for a broad set of starting orientations in the same aspect. These results are in agreement with those reported by Lowe [76]. We conclude that large numbers of fine aspects are not needed for modeling. The experimental results also

demonstrate that smooth objects can be handled and some occlusion can be tolerated. Our experiments with superquadrics indicate that the alignment method can be used with most conventional CAD systems.

Since the Newton's method with Levenberg-Marquardt minimization is used to iteratively diminish the error between the observed edgemap and the fitted edgemap, sensitivity of a parameter to the error influences the goodness of the fit of the parameter. Normally, the more sensitive the parameter is, the more accurate the parameter fit would be. It is obvious that the scaling factor s and the translation vector (t_x, t_y) are very sensitive because a small perturbation in any of these three parameters would create a significant error between the fitted edgemap and the observed edgemap. Thus, as shown in our experimental results, the estimates of these three parameters tend to be more accurate than any others.

Finally, we have shown mathematically that aligning silhouettes alone provides inaccurate pose parameters for rotation for objects which are somewhat symmetric about the rotation axis. The same analysis shows that the rotation parameters are much more sensitive to location of internal edges in the images. This principle is supported by our experimental results. Thus, we conclude that including internal edges in alignment is essential for an accurate pose estimation of many curved objects. This result is general in the sense that it can be used by other published iterative methods to reduce inefficient wandering in the parameter space as well as inaccurate orientation in convergence results.

Chapter 4

Indexing to Model Aspects

4.1 Introduction

Model-based object recognition requires that sensed data from an object be matched to pre-stored model data. In the alignment paradigm, minimal sets of features are used to form correspondence hypotheses between an image and a pre-stored object model. For each such correspondence, a transformation is computed which brings the model features into alignment with the image features, and model presence is verified by back-projecting other model features into the image and searching for image features at those locations. The entire process can be very expensive due to the possible combinatoric correspondence of feature sets for generating hypotheses and the exhaustive search at each verification.

The most straightforward way for object recognition from a model database is by linear search. That is, one can apply the alignment paradigm to each object in the database in a sequential fashion, and the object model that best aligns the observed

image features is recognized. With this search strategy, the recognition time increases linearly with the number of possible object models in the database. As the model database size grows, the linear search strategy decreases in viability, some better means of selecting object models are needed.

Indexing is one way to alleviate the computational burden. It is a two-stage process. At compile time, feature sets derived from object models are encoded as indexing vectors corresponding to entries in some appropriate data structure, such as a hash table or tree, with pointers back to the corresponding models. This provides the essential mechanism for indexing and fast retrieval. At run-time, feature sets derived from the image form index vectors for fast access of pre-stored models. Thus, correspondence hypotheses are recovered without resorting to comparison of all pairs of model/image feature sets.

Although correspondence hypotheses can be quickly recovered through indexing, there may still exist some ambiguity among the indexed hypotheses. This situation arises when a single index vector derived from the sensed image retrieves many competing match hypotheses. This problem can be overcome by providing the indexing scheme with some additional information, such as the probabilistic peaking effect used in [87] or the indexing function used in [10], to determine which hypothesis is more likely to give a correct interpretation for the data. This additional information not only helps in grouping hypotheses into consistent clusters, but also provides a mechanism to rank the match hypotheses before sending them to the verification stage. These rankings will in general mean that correct interpretations are likely to be verified first, resulting in an improved over-all efficiency in the recognition process.

Our approach toward the indexing task within the alignment paradigm is as follows. As described in Chapter 2, we model a 3D object by a collection of $2\frac{1}{2}$D views (called model aspects) made from 5 local images taken by rotating the viewpoint up, down, left and right of a central viewpoint (see Figure 2.2). The 3D information is made from the central edgemap of each model aspect and is used for pose estimation. The silhouette of the central image is extracted and segmented into *parts* whose invariant features are derived for indexing. Thus, each model aspect consists of a $2\frac{1}{2}$D edgemap for pose estimation (i.e., verification) and part invariant features for indexing. Both model building and index building phases can be processed off-line. During the recognition phase, we first extract the part invariant features from the sensed image to index into the model database for candidate models and poses. We then apply some heuristics to determine the order of the hypotheses for verification. Finally, verification is carried out by fitting the $2\frac{1}{2}$D edgemap of each candidate model to the observed edgemap (i.e., pose estimation), as done in Chapter 3. The result of the fitting is then used to refute or accept the model hypothesis. Once a correct model is found, the verification procedure is terminated. A diagram of the proposed indexing scheme is given in Figure 4.1.

This chapter is organized with the next section reviewing some existing techniques for indexing 3D models from 2D images. Section 4.3 presents the general framework for the proposed indexing scheme. Section 4.4 shows experimental results. The last section summarizes the proposed approach.

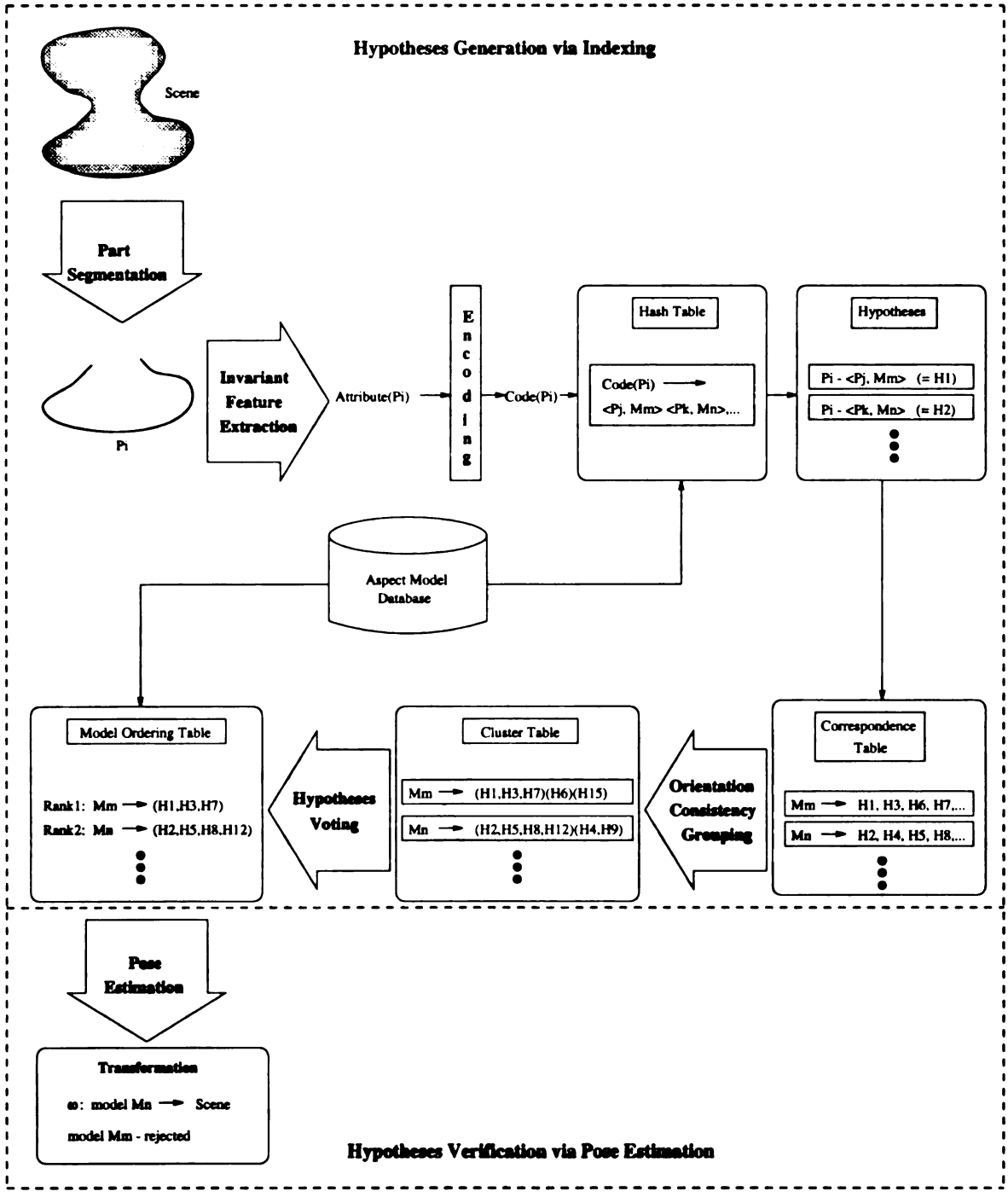


Figure 4.1: The computational paradigm of the proposed indexing scheme.

4.2 Related Background

We are interested in indexing to 3D models from single 2D intensity images. Indexing techniques are becoming widespread in the field of computer vision for solving the recognition problem and the model/sensory feature correspondence problem. Indexing involves extracting an over-constraining indexing coordinates from a group of image features and then interpreting these features through a pre-compiled lookup table [115]. Much work in the past on indexing has dealt with the easier problems of 2D models (*e.g.*, [101, 65]), flat 3D models (*e.g.*, [72, 96]), or 3D models in which more information is available, such as with range data (*e.g.*, [43, 102]), or multiple images [81].

When compiling an index, the most efficient storage possible can be achieved by using view-invariant shape descriptors that are unaffected by object pose, perspective projection, and the intrinsic parameters of the camera [44]. For each underlying model feature set, only a single index vector is sufficient to identify the object. Thus, view-invariant descriptors are ideal candidates to be used to directly index into the model database for fast object identification. However, it has been shown [22, 33] that there are no *general-case* view-invariant features for any number of 3D points under common of projection model (perspective, weak perspective, or parallel). As a result, most work in indexing using invariant descriptors has been restricted to 2D objects [23, 101] or 3D planar objects [72, 85, 96], or to some special-case invariants, which are only view invariant under special configurations of 3D points or set of views [22, 44, 110]. For example, assuming that the object points are planar and

the projection is approximated by the weak perspective projection model, Lamdan [72] was able to extract *affine coordinates* which are invariant to view. Burns *et al.* [22] have defined and discussed affine coordinates and other special-case invariants under weak perspective projection model. Forsyth *et al.* [44] have also assumed a planar surface and have shown that an invariant, depending on the coefficients of the conics, can be obtained by fitting two conics to two image curves projected from this surface. Thus, as pointed out in [22], "invariants that do not require any restrictions are impossible, and recognition systems using unrestricted invariants can not be expected to be as effective as those using special-case schemes".

One way to circumvent the above problem is to represent a 3D object by multiple 2D views. Each view is treated as a model, then the indexing problem is reduced to the problem of indexing a 2D object using 2D invariants, which have been successfully used by many researchers [19, 33, 61, 109]. For example, Clemens and Jacobs [33] grouped four point features as a structure for indexing, three of which were used to generate the object pose and one of which was used to check if the pose consistency is preserved. Groups that violate the pose constraint were discarded, resulting in a considerable amount of reduction in time and space in indexing. Jacobs [61] provided a more space-efficient construction reducing 2D sheets to two 1D lines, each embedded in a 2D space. Thompson and Mundy [109] grouped pairs of vertices to retrieve hypothetical viewing transformations for their models. Due to the lack of specificity in the indices, a voting scheme was used to integrate the noisy pieces of information into meaningful hypotheses.

Breuel [19] argues that to effectively support indexing, one needs a representation

of features that is invariant under 2D affine transformation (translation, rotation and scaling), under occlusion of features, and under the addition of spurious features. However, using this kind of indexing scheme for 3D object exacerbates the recognition time by vastly expanding the model database. This raises the issue of determining how many distinctive views of an object model should be stored in the database.

The aforementioned methods discretize the indices and fill the hash table with the quantized values. Attempting to cover all possible views of a feature set by filling in a discrete look-up table is a formidable task, especially in high-dimensional indexing spaces. One approach to this problem is by tessellating the viewsphere into a large fixed number of views and then integrating neighboring views which share same feature sets into aspects [36]. Another promising approach is by using some interpolating functions, such as Radial Basis Functions (RBF) proposed by Poggio and Edelman [90], to approximate the regions between sample views. Each distinct interpolated feature set, covering a small region of views, is filled into the discrete look-up table. In this way, a full range views of an object can be modeled from a relatively small number of views. Beis and Lowe [10] used a learning method to generate indexing functions which interpolate between the index vectors sampled at various views about the stored models. The indexing functions provide probability distributions describing the possible interpretations of each index value. The resulting probabilities are used to rank the match hypotheses for verification.

4.3 General Framework for Indexing

In this section, we develop the framework for the proposed indexing scheme (see Figure 4.1). From the original problem assumptions, the following limits apply. First, the partitioning rule, proposed by Hoffman and Richards [54], is used to segment 2D shape into parts. Second, part invariant features are derived for indexing; the indexing is carried out via hashing. Third, parts stemming from the same model are integrated together to form groups of consistent hypotheses. Finally, four voting schemes are proposed to rank the hypotheses in order to reduce the number of verifications required during the recognition.

4.3.1 Part Segmentation

The notion of recognition based on "parts" has become increasingly popular. Perhaps the most compelling support for this idea is based on recognition in the presence of occlusion. Partial occlusion of the object renders global shape descriptors ineffective for indexing or recognition. In the alignment paradigm, verification or refutation of a candidate model (i.e., pose estimation) is done in a global manner which considers portions of model features and portions of sensed features. Occluding objects and artifacts in feature extraction may degrade the match value, but the alignment method is still effective because fragments of matching data can be integrated into a single global model-pose hypothesis. The indexing phase, however, must operate without a hypothesis. Partitioning a sensed shape into parts seems inevitable for this phase of object recognition. We use the central silhouette of an aspect model to index

the aspect, thus, views within the aspect should have more or less same indexing features. The success of the indexing phase counts on a reliable partitioning scheme that should be *invariant* to similarity transformation, *robust* to local deformation, and *stable* with global transformation. A significant aspect of a partitioning scheme is the way decomposed parts behave under various changes that may arise in the visual projection. First, changes in shape due to translation, rotation, and scaling of the 2D shape must not effect the part structure within the shape. Second, changes are localized to certain portions of the shape. It is natural to demand that local deformations of the shape should not effect those parts remote to the change. Third, changes arising from global deformations, such as changes in viewing direction or viewpoint within a model aspect, or deformation of the boundary due to noise and spurious data, should only have a small effect on the 2D shape and its part structure.

How should parts of a shape be computed? Several approaches in the past suggest either the decomposition of its interior region or the segmentation of its boundary. Region-based parts include a description based on the combination of primitives such as generalized cylinders [14, 79], and superquadrics [5, 89], or the neck-based and limb-based description proposed in [99]. In contrast, contour-based segmentations partition the boundary at zero-crossings of curvature [83] or at high-curvature points [45]. Primitive-based boundary partitioning schemes approximate the boundary as a combination of primitives such as constant curvature segments [119] or using a curvature primal sketch (e.g., cranks, bends, bumps, and ends) [3]. A significant departure from the traditional boundary-based techniques is the method proposed by Hoffman and Richards [53] who advocate that ***part decomposition should precede***

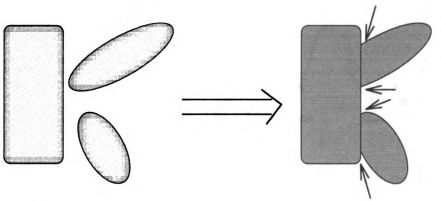


Figure 4.2: Hoffman and Richard's theory of curve partitioning: joining parts generally provides concavities in the resulting silhouette.

part description. In contrast to primitive-based approaches, they propose a theory of parts which depends not on the shape of parts, as described by primitives, but rather on general principles underlying their formations or "regularities of nature". The *transversality* principle is an example of such a regularity, asserting that "when two 3D entities are combined together to form a composite object, concavities are created at the join". This principle provides very general rules for segmenting either 2D or 3D objects into parts [54]. Part boundaries are naturally delimited by the concavities. In the case of 2D plane curves or silhouettes, these concavities, as indicated by the small arrows in Figure 4.2, appear as cusps and occur at minima of negative curvature. Thus, our rule for partitioning 2D shapes is as follows:

> **Partitioning Rule:** *segment a curve at concave cusps (or minima of negative curvature) to break the shape into its parts.*

This partitioning rule leads to a representation of the shape boundary based on *codons* [53]. Codons are formed by partitioning curves at minima of curvature; the maxima and zeros of curvature are used to describe the shape of each segment, result-

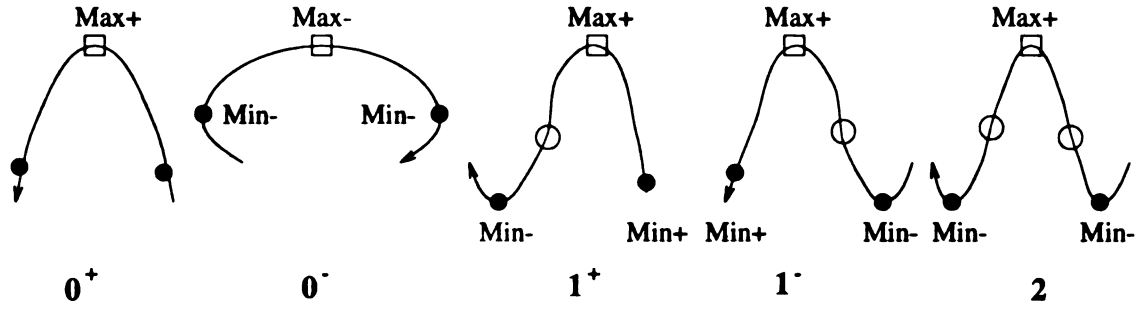
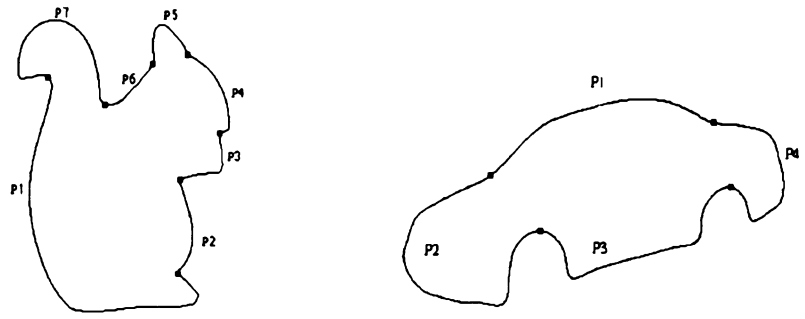


Figure 4.3: The primitive codon types proposed by Hoffman and Richard.

ing in five types of codon: 0^+, 0^-, 1^+, 1^-, and 2, shown in Figure 4.3, where maxima of curvature are shown as □ and labeled Max+ if positive or Max- if negative, minima of curvatures as • and labeled as Min+ if positive or Min- if negative, and zero of curvatures as ○; the arrow shows the traversal of the direction; if the direction of rotation is counterclockwise, the curvature is positive; otherwise, it is negative; the figure is placed to the left of the direction of traversal, and the background is to the right. There is one more codon that describes the degenerate case of a straight line, i.e., segment with an infinite number of points of zero curvature. Rosin [94] has extended the five standard codon descriptors to include all cases where a codon is neighbored by a straight segment or the case where the curve is not necessarily closed.

The codon representation is very sensitive to noise due to the computation of curvature, which can cause many spurious curvature extrema. To suppress noise and keep gross shape, the curves should be smoothed when computing curvature. Mokhtarian and Mackworth have shown that cusps are formed when there is a self-intersecting loop on the contour, and the curve is smoothed with a large-scale Gaussian filter [82]. Fortunately, a self-intersecting loop cannot occur on the object silhouette, and thus, no phantom part boundary would be created by curve smoothing. However,

the codon representation is still unstable, preventing it from being a reliable part descriptor by itself.



(a) (b)

Figure 4.4: Segmentation of object silhouette into parts. (a) Boundary of squirrel carving segmented into 7 parts. (b) Boundary of an automobile (Taurus) segmented into 4 parts.

Examples

We illustrate our part segmentation scheme on several examples of man-made and sculptured objects. We use a Gaussian filter with a kernel width of 55 pixels to smooth a curve before the part segmentation scheme is applied to the curve. The results of the segmentation scheme applied to the silhouettes of a 3D wood carving of a squirrel and a model car are shown in Figure 4.4. The segmentation of the car in Figure 4.4(b) might be considered by some to be unnatural. Were the wheels extracted along with the body, the situation would change; however, we have had difficulty in reliably extracting the wheels. In theory, the part representation is immune to rotation, translation, and scaling. Although occasionally the representation may be affected by

noise and quantization (i.e., some parts may be slightly distorted), the segmentation technique provides appropriate representations in practice. Moreover, because the computation of curvature is local, occlusion of some parts of the object does not affect other parts that are within a small distance from the occlusion. More examples are given in Figure 4.5 to demonstrate the generality of our part segmentation algorithm; the decompositions are natural for a variety of objects.

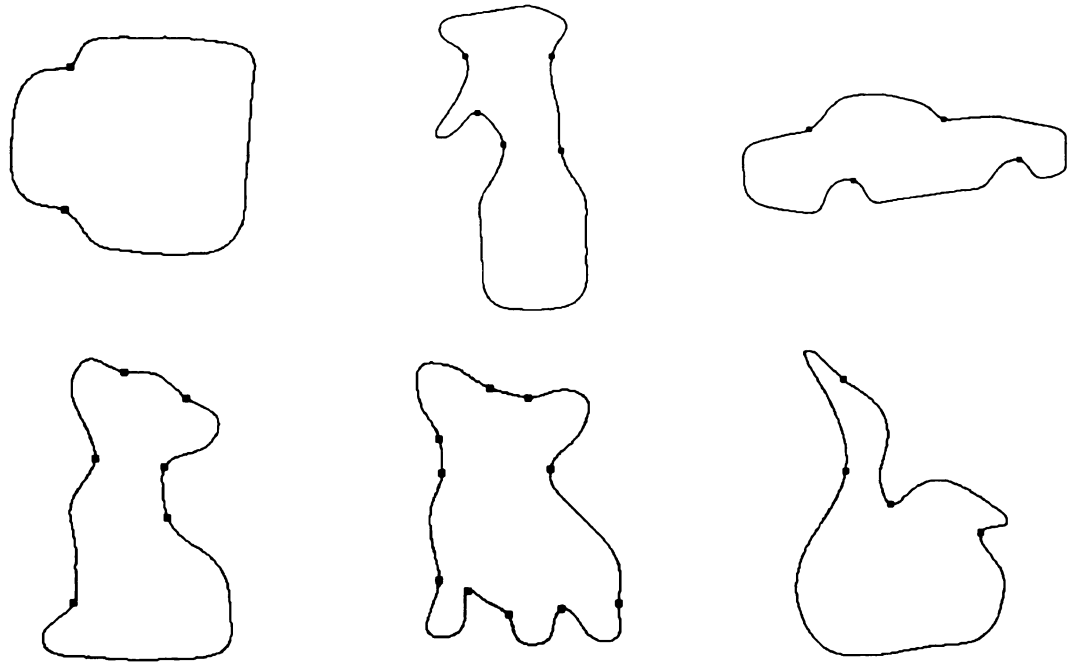


Figure 4.5: Examples of part segmentation on a variety of shapes.

Figure 4.6 demonstrates the robustness of the part segmentation scheme in the presence of occlusion. Portions of occluded and unoccluded versions of the objects with identical shapes have identical parts, as required by the robustness criterion. Figure 4.7 illustrates the robustness of the process under noise and scaling. The top row of Figure 4.7 shows the contours scaled 100%, 75%, 50%, and 25% from left to

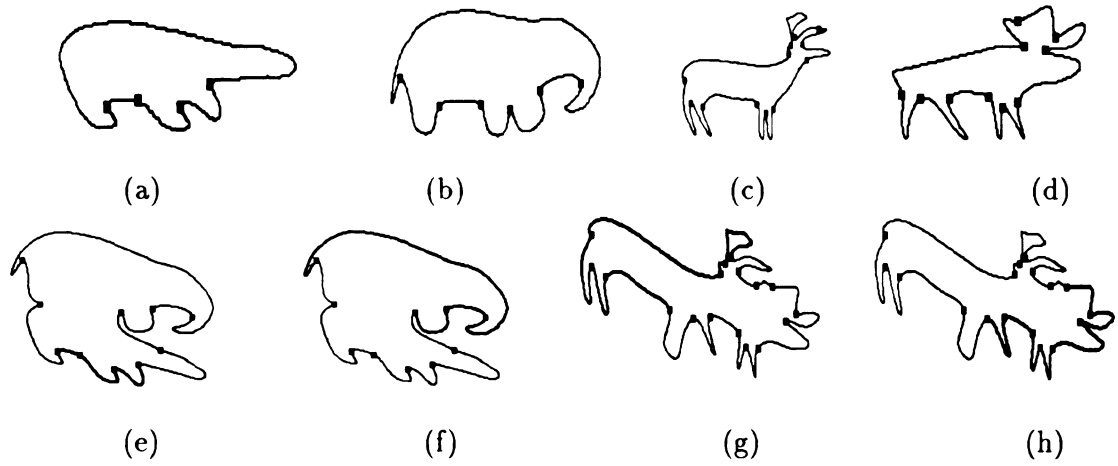


Figure 4.6: Part evidence survives occlusion. (a)–(d) Parts extracted from silhouette images of single bear, elephant, deer, and moose, respectively. (e)–(h) Thicker lines indicate parts used for generating hypotheses for bear, elephant, deer, and moose, respectively.

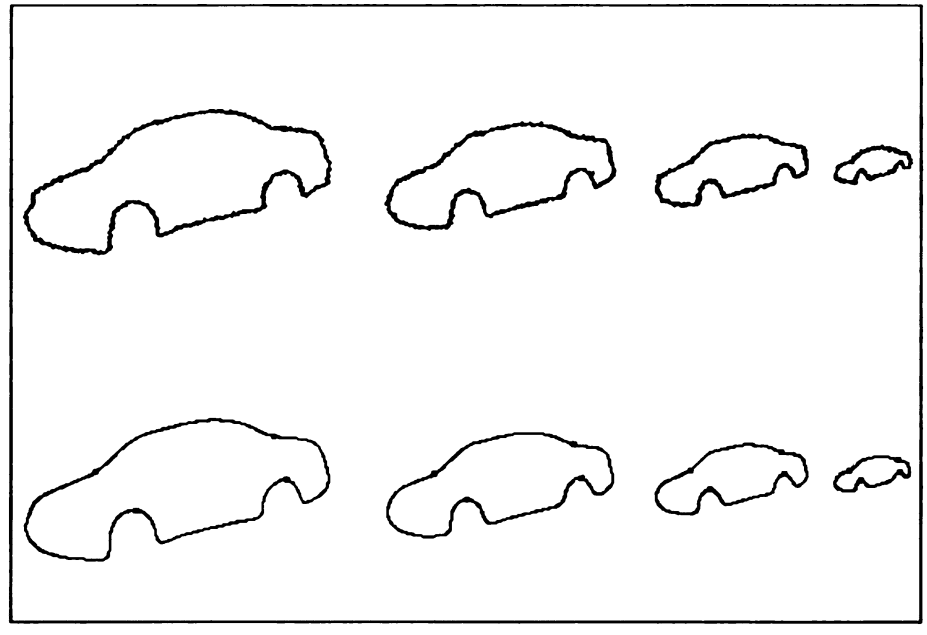


Figure 4.7: Part segmentation survives noise and scaling. The black dots indicate the boundaries of segmented parts.

right, respectively. Gaussian noise with 0 mean and 2 pixels standard deviation was added to the contours. The segmentation remains stable as shown in the bottom row of Figure 4.7.

Figure 4.8 illustrates the robustness of the part segmentation scheme under global transformation due to view variations. The segmented parts of two sculptured objects, pencil sharpener and phone, are shown in Figure 4.8 (a) and (b) respectively. The extreme views of these two model aspects (i.e., the object is rotated 10° around the vertical axis to the left) were used for part segmentation to generate parts as shown in Figure 4.8 (c) and (d). This is the worst case within a model aspect. The parts near the pencil sharpener handle are distorted due to self-occlusion while the rest of parts remain intact. Although the part segmentation scheme produces exact parts for the phone, their shapes seem to vary a little. This means that the indexing scheme must compensate for the part variation by loosing the tolerances of indexing features extracted from parts. In Figure 4.8(e) and (f), we show the segmented contours of (c) and (d) under 2D transformation (rotation and translation). Figure 4.8 indicates that our part representation is invariant under 2D transformation but slightly variant under 3D transformation due to self-occlusion and the change of views (however, some of the parts retain similar structure).

4.3.2 Part Indexing

Some important design criteria for an indexing scheme are: (1) the indexing primitives should be invariant under translation, rotation and uniform scaling; (2) the

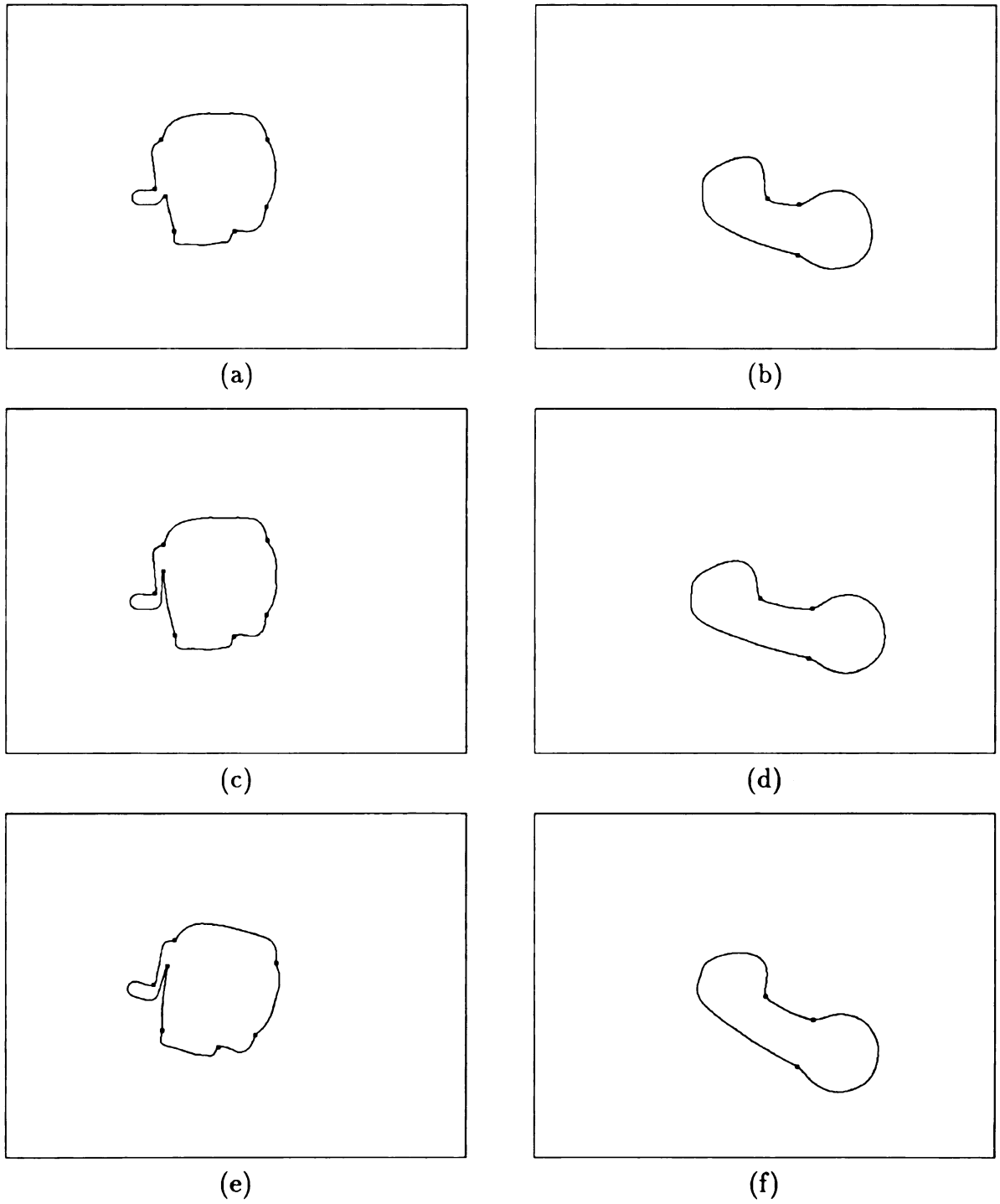


Figure 4.8: Part segmentation survives 2D transformation and 3D view distortion.

indexing scheme should be robust under partial occlusion of the object; (3) the indexing scheme should be able to handle spurious and noisy data. All these three criteria are met by our part segmentation scheme. Under this segmentation scheme, object boundaries can be reliably segmented into parts that are generally invariant to rotation, translation, and scaling. In addition, the degradation of the representation due to occlusion is roughly proportional to the amount of occlusion. In this section, we derive invariant *quantitative attributes* from the parts and show how these attributes can be used for indexing.

Indexing Attributes and Invariants

Given a part (curve) C, let $\mathbf{p_i} = (x_i, y_i), i = 0, 1, \cdots, N-1$ be the points of C. If C is not closed, then it can be closed by appending the first point ($\mathbf{p_N} = \mathbf{p_0}$) so that C encloses a 2D region. This allows us to compute region attributes from C. We define the following attributes to encode the part: assuming that no portion of C is occluded, these attributes satisfy the criteria for indexing.

Compactness: Let A and P be the enclosed area and perimeter of C respectively. The compactness of the part, $\varsigma(C)$, is defined as P^2/A where

$$P = \sum_{i=0}^{N-1} \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2}$$

$$A = \frac{1}{2} \sum_{i=0}^{N-1} \begin{vmatrix} x_i & y_i \\ x_{i+1} & y_{i+1} \end{vmatrix}$$

where $|\cdot|$ denotes the determinant. The quantity ς has been shown to be invariant to translation, rotation, and scaling [6, 74].

Roundness: the roundness of C, $\tau(C)$, is the sum of the absolute curvatures along C, i.e., $\sum_{i=0}^{N-1} |\kappa_i|$ where κ_i is the curvature at point $\mathbf{p_i}$. The quantity τ can be shown to be invariant to scaling, and the curvature κ is also known to be invariant to rotation and translation.

Skewness: Let $\mathbf{p_{N/2}}$ be the midpoint of the part and $\mathbf{p_m}$ the midpoint of the line connecting $\mathbf{p_0}$ and $\mathbf{p_{N-1}}$. The skewness, of the part, $\rho(C)$ is defined as

$$\rho(C) = \cos^{-1} \frac{\mathbf{p_0\vec{p}_{N-1}} \cdot \mathbf{p_m\vec{p}_{N/2}}}{\|\mathbf{p_0\vec{p}_{N-1}}\| \|\mathbf{p_m\vec{p}_{N/2}}\|}$$

Moment invariants: modified moments using only the curve boundary were defined in [26] as follows

$$m_{pq} = \int_C x^p y^q ds, \quad \text{for } p, q = 0, 1, 2, \cdots$$

where $\int_C$ is a line integral along the curve C and $ds = \sqrt{(dx)^2 + (dy)^2}$. The modified central moments are thus similarly defined as

$$\mu_{pq} = \int_C (x - \bar{x})^p (y - \bar{y})^q ds, \tag{4.1}$$

where $\bar{x} = m_{10}/m_{00}$ and $\bar{y} = m_{01}/m_{00}$. For a digital curve C, Eq. (4.1) becomes

$$\mu_{pq} = \sum_{(x,y)\in C} (x - \bar{x})^p (y - \bar{y})^q. \tag{4.2}$$

Normalizing the central moments, we obtain

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{p+q+1}} \text{ for } p + q = 2, 3, \cdots$$

These normalized central moments are used to derive the following:

$$\begin{aligned}
\phi_1 &= \eta_{20} + \eta_{02} \\
\phi_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
\phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (\eta_{03} - 3\eta_{21})^2 \\
\phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{03} + \eta_{21})^2 \\
\phi_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03}) \\
&\quad \times (\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
\phi_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\
\phi_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{12} - \eta_{30}) \\
&\quad \times (\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2].
\end{aligned}$$

Hu [58] showed that the quantities, $\phi_i, 1 \leq i \leq 7$, are invariant to translation, rotation, and scaling. Since the quantities $\phi_i, 2 \leq i \leq 7$ are very small and have little discriminating power, we only use ϕ_1 for indexing.

Convexity: The codon representation is susceptible to spurious extrema resulting from noise, preventing it from being a reliable part descriptor by itself. However, it still can be used to determine whether a part is concave or convex: let $\sigma(C)$ denote this shape property of a part C. Obviously, σ can provide some discriminating power for part recognition.

We encode each part C using the above attributes quantized in some appropriate manner. All encoded parts serve as indexing primitives to search a hash table for model hypotheses. The endpoints of each part, $\mathbf{p_0}$ and $\mathbf{p_{N-1}}$, encode the pose of the part. A third point, such as the point of maximum absolute curvature encodes how the contour lies relative to the endpoints. Note that for each pair of the corresponding scene part and model part, these three point correspondences is sufficient to determine the scale factor of the object size and the roll angle of the image plane as mentioned in Chapters 2 and 3. If more than one pair exists, the roll angle can be derived in a least-square error minimization fashion. The aforementioned (variant) attributes can be used in grouping consistent hypotheses (indexed from parts) into clusters and also provide an approximate alignment between a sensed part and a model part similar to it, and thus provide an initial pose hypothesis for the pose estimation algorithm (i.e., the Newton's method with Levenberg-Marquardt minimization) which searches for a global alignment using all parts.

Sketch of Indexing Scheme

Our indexing scheme discretizes the indices and fills the hash table with quantized values. The advantage is that the run-time indexing step can be done in constant

time. The disadvantage is that the performance of indexing may be degraded when many part hypotheses collide in one hash bin. This arises due to the standard way these methods have dealt with noisy images. During table construction, an entry is made in each hash bin within an error radius ϵ of the actual stored model index value. Then all relevant hypotheses can still be accessed by looking in just one bin. The number of collisions in a bin depends on the degree of specificity of the bin. More specificity means finer bin divisions, but the probability of false dismissal increases accordingly. On the other hand, less specificity means coarse bin divisions, and the probability of false alarm increases. The only way to deal with this noise issue is, for each part indexing attribute, to search the ϵ-volume of index space around the index (i.e., range search) at run-time, and then to form the union of the retrieved hypotheses. This certainly reduces the original advantage of the approach. The sketch of our indexing scheme using range search is given next (see Figure 4.1 for clarity).

Each $2\frac{1}{2}$D edgemap of a model aspect requires a few kilobytes of storage while the attributes of each part indexing to the model aspect requires a few dozen bytes. Each model part results in one record being inserted into the hash table. Using the parts from the sensed image, the index step produces a set of hypotheses for those models needing to be verified. The hypotheses are then grouped together based on their model candidates. A good indexing scheme should index into a set of N models in $O(logN)$ time or better, provided that there are no more than $logN$ relevant hypotheses because the system cannot afford to refute too many hypotheses. Given part representation $(\sigma, \varsigma, \tau, \rho, \phi_1)$, the indexing scheme performs a range search using $(\sigma, \varsigma \pm \epsilon_\varsigma, \tau \pm \epsilon_\tau, \rho \pm \epsilon_\rho, \phi_1 \pm \epsilon_{\phi_1})$, where ϵ_x is the error tolerance of attribute x, for

the parts having attributes within this range in a database. The tolerance allows for errors due to quantization, noise, and view variations of shape within a model aspect (recall each model aspect is represented by the attributes extracted from the *central* viewpoint of that aspect). The range search can be implemented using a $k-d$ tree structure with a worst-case running time of $O(k \cdot N^{1-1/k})$ [95]. We implement the range search using a hashing scheme. For each part attribute, we encode its range and then use each encoded index within this range to hash into an open hash table with b buckets. The average-case search time would be $O(N/b)$ for each part. Making b a significant fraction of N brings the search time down to a constant for most searches.

4.3.3 Hypothesis Grouping

For indexing to be effective, it is important that some data-driven grouping mechanism produce sets of parts likely to come from a single object [75]. Multiple scene parts drawn from the same observed object should yield similar relative orientation with parts bound to the same model candidate (e.g. poses should cluster). Scene parts, drawn from different objects but indexed to the same model candidate or drawn from the same object but indexed to the same wrong model candidate, are not likely to have consistent orientations relative to their corresponding parts in the model candidate. Thus, the correct model candidate will tend to have one large group of hypotheses while the wrong model candidates will tend to have several small groups of hypotheses. Consequently, when hypotheses ordering is enforced on the groups of consistent hypotheses, the correct model candidate will tend to have a better rank

(*rank* is defined as the order of the model candidates in the verification phase).

Let $\{H : p_i \rightarrow (p'_i, M_m)\}$ be the hypothesis that the scene part p_i corresponds to a part p'_i in the model candidate M_m. Let $\mathcal{H} = \{H_1, H_2, \cdots, H_n\}$ be the n hypotheses generated from the indexing of all the scene parts. We divide these n hypotheses according to the models to which the scene parts index and store them into a correspondence table where the model M_m serves as a key and the hypotheses $\mathcal{H}_m = \{H_{m_1}, \cdots, H_{m_n}\}$ (with $\mathcal{H}_m \subseteq \mathcal{H}$) as entries. For each model candidate M_m with a hypothesis list $\mathcal{H}_m$, we group the consistent hypotheses based on the relative orientation between the two corresponding parts within a hypothesis as follows. Let (x_1, y_1) and (x_n, y_n) be the two end points of a part p. The orientation of p is defined by the angle $\theta = \tan^{-1} \frac{x_n - x_1}{y_n - y_1}$. Let $\{H_i : (p_i \rightarrow (p'_i, M_m)\}$ and $\{H_j : (p_j \rightarrow (p'_j, M_m)\}$ be two hypotheses from the hypothesis list of the same model candidate with the orientations, θ_i, θ'_i, θ_j, and θ'_j for parts p_i, p'_i, p_j, and p'_j respectively. Then two hypotheses H_i and H_j are said to be consistent if

$$|(\theta_i - \theta_j) - (\theta'_i - \theta'_j)| \leq \epsilon_\theta, \tag{4.3}$$

where ϵ_θ denotes some bounds on the tolerance of orientation difference. This consistency grouping of hypotheses may seem to be quite complex because we need to check every hypothesis against all others, but in fact, we do not need to do so. We can rearrange Eq.(4.3) as $|(\theta_i - \theta'_i) - (\theta_j - \theta'_j)| \leq \epsilon_\theta$ and quantize the difference of orientation using $2\epsilon_\theta$ as a unit into cells in the range of $[-2\pi, 2\pi]$, then, H_i and H_j are said to be consistent if $\theta_i - \theta'_i$ and $\theta_j - \theta'_j$ fall into the same cell. In this way,

hypotheses in the same cells are grouped together. Thus, hypotheses in $\mathcal{H}_m$ for model M_m are divided into some disjoint groups $\mathcal{H}_{m_1}, \cdots, \mathcal{H}_{m_l}$ with $\mathcal{H}_{m_i} \cap \mathcal{H}_{m_j} = \emptyset, \forall i \neq j$.

4.3.4 Hypothesis Ordering

There are a few problems associated with our indexing scheme described above, especially when a large database is considered. The model hypotheses are non-uniformly distributed in the hash table, resulting in some entries with a large number of model hypotheses. This situation occurs frequently when the database contains many objects with similar parts. To minimize the recognition time, our indexing scheme should exploit prior knowledge to reduce the number of verifications during the recognition because the verification procedure is computationally expensive. This is accomplished by verifying the most likely model hypothesis first and then terminating the verification procedure once a correct model is identified. Thus, our indexing scheme relies heavily on the voting scheme for hypothesis ordering to reduce the recognition time. Four voting schemes are studied below to make efficient use of prior knowledge for hypothesis ordering.

Majority Voting

Majority voting is the most common voting scheme: the votes are accumulated for model hypotheses on the basis of the presence of their parts in the scene. The basic idea behind this voting scheme is based on the fact that multiple parts from the same scene object should support the same model candidate. Thus, if we accumulate votes from all scene parts and rank the model hypotheses according to the total

votes received, then the model hypotheses with the largest number of votes are good starting points for the verification process.

Let $\mathcal{M} = \{M_1, M_2, \cdots, M_N\}$ be a set of all models in the database, and $\mathcal{P} = \{p_1, p_2, \cdots, p_m\}$ a collection of scene parts. Let $\mathcal{MH} = \{(p_i, M_k), i = 1, \cdots, m, k = i_1, \cdots, i_n\}$ represent the model-hypothesis list retrieved from the hash table by every scene part $p_i \in \mathcal{P}$ where $M_k, k = i_1, \cdots, i_n$ are model candidates. Define a characteristic function $\delta(p, M)$ as

$$\delta(p, M) = \begin{cases} 1 & \text{if } p \text{ is a part of } M \\ 0 & \text{otherwise} \end{cases}$$

where p is the part identifier and M the model identifier. Then the number of votes that model M receives is

$$R_{MV}(M) = \sum_{i=1}^{m} \delta(p_i, M).$$

where $R_{MV}(M)$ denotes the score of the model hypothesis M obtained via the majority voting scheme. The most likely candidate models M^* have the largest number of votes $M^* = \arg\max_M R_{MV}(M)$.

Bayesian Voting Using Part Occurrence

The prior knowledge incorporated in the majority voting scheme described above assumes that each model candidate, in the model-hypothesis list indexed by a scene part, has the same likelihood of being identified by that scene part. This assumption may not reflect the real situation where some model candidates are more likely to

appear in the scene than others. Thus, given a scene part, the goal is to identify the model candidate which most likely accounts for the presence of this scene part. To achieve this goal, we adopt a decision-theoretic approach using a Bayesian framework where the measure of the discriminatory power of a part for a model is defined in terms of posterior probability (similar to the Bayesian framework in [120]).

Let $P(M_k)$ be the *prior* probability that model M_k is observed, and $P(p_i|M_k)$ be the likelihood function where $P(p_i|M_i) > P(p_i|M_j)$ means that part p_i is more likely to be observed when model M_i is seen than when model M_j is seen. Let $P(M_k|p_i)$ be the *posterior* probability that reflects the updated belief that model M_k appears in the scene after the part p_i is observed.

In order to derive the posterior probability $P(M_k|p_i)$, we need to estimate probabilities $P(p_i|M_k)$ and $P(M_k)$. If we have no other information, we assume $\hat{P}(M_k)$ is the same for every model in $\mathcal{M}$. Once a certain part, p_i, is decided to be used for indexing for model candidates, we compute the likelihood function $\hat{P}(p_i|M_k)$ by counting the number of occurrences of part p_i in the model M_k:

$$\hat{P}(p_i|M_k) = \frac{O(p_i, M_k)}{\sum_{\forall p_i \in M_k} O(p_i, M_k)} \tag{4.4}$$

where $O(p_i, M_k)$ denotes the number of occurrence for a particular part p_i in model M_k. Note that $\hat{P}(p_i|M_k) = 0$ if p_i does not exist in the model M_k. Let $\mathcal{M}_f \in \mathcal{M}$ be the set of model candidates retrieved from the hash table by using p_i as the index.

The posterior probability $\hat{P}(M_k|p_i)$ is then computed as

$$\hat{P}(M_k|p_i) = \frac{\hat{P}(M_k)\hat{P}(p_i|M_k)}{\sum_{\forall M_k \in \mathcal{M}_f} \hat{P}(M_k)\hat{P}(p_i|M_k)}. \tag{4.5}$$

This formula is used to determine the distribution of the posterior probabilities among the models in $\mathcal{M}_f$ indexed by p_i. That is

$$\sum_{M_k \in \mathcal{M}_f} \hat{P}(M_k|p_i) = 1.0$$

Define $\hat{P}(M_k|p_i)$ as *the indexing power of* p_i *for the model* M_k. If $\hat{P}(M_k|p_i) = 1.0$, this means that p_i is unique to the model M_k. In other words, if a part detected in the scene corresponds to p_i, it is certain that the model M_k is in the scene. On the other hand, if the indexing power of a part is not 1.0, several models containing the part may be in the scene. Suppose M_i and M_j are two of the model candidates indexed by p_i, then $[\hat{P}(M_i|p_i) > \hat{P}(M_j|p_i)]$ indicates the belief that model object M_i is more likely to appear in the scene. This indexing power is a decision-theoretic measure of saliency of a part for identifying a model under the Bayesian framework whereas a saliency measure based on population [43] is somewhat *ad hoc*. Once the posterior probability $\hat{P}(M_k|p_i)$ is computed for every model candidate $M_k \in \mathcal{M}_f$ and every scene part $p_i \in \mathcal{P}$, we can rank the model hypotheses $M \in \mathcal{MH}$ by

$$R_{BVO}(M) = \sum_{i=i}^{m} \hat{P}(M|p_i)\delta(p_i, M) \tag{4.6}$$

where $R_{BVO}(M)$ denotes the score of the model hypothesis M obtained via the

Bayesian voting scheme using part occurrence.

Bayesian Voting Using Visual Relevance

The estimated prior probability $\hat{P}(p_i|M_k)$ in Eq.(4.4) assumes that parts belonging to the same model have equal probability (i.e., $\forall p_i \in M_k, i = 1, \cdots, n, \hat{P}(p_i|M_k) = \frac{1}{n}$). This estimation overlooks the fact that some parts may be more likely to identify the object than other parts belonging to the same model. For example, a model M is composed of two parts p_1 and p_2 with p_1 occupying 10% of the contour and p_2 90%. Identifying p_1 in the scene does not give strong belief that model M is present in the scene; however, if p_2 is observed in the scene, then we have much higher confidence that model M is present in the scene. Thus, we assign the prior probability to parts of the same model based on their relative visual relevance. This measure uses the length of the part normalized with respect to the size of the model. Then the likelihood function in Eq.(4.4) is changed to

$$\hat{P}(p_i|M_k) = \frac{L(p_i, M_k)}{\sum_{\forall p_i \in M_k} L(p_i, M_k)} \tag{4.7}$$

where $L(p_i, M_k)$ denotes the length of part p_i occurring in model M_k. The posterior probability $\hat{P}(M|p_i)$ is given by the same formula as in Eq.(4.5).

Note that the weights $W(p_i)$ of scene parts $p_i, i = 1, \cdots, m$ were assigned equally in BVO, i.e., $W(p_i) = 1/m$, because parts were equally likely to occur in the scene, and thus, were not considered in Eq.(4.6). Here we assign the weights of scene parts based on their relative length, i.e., $W(p_i) = Q(p_i)/\sum_{i=1}^{m} Q(p_i)$ where $Q(p_i)$ denotes

the length of p_i. Then the ordering of model hypotheses is based on

$$R_{BVVR}(M) = \sum_{i=1}^{m} \hat{P}(M|p_i)W(p_i)\delta(p_i, M) \tag{4.8}$$

where $R_{BVVR}(M)$ denotes the score of the model hypothesis M obtained via the Bayesian voting scheme using visual relevance. The justification for incorporating the weight of scene parts into the hypothesis ordering process is the same as the explanation in the example given above.

The Hybrid Voting Scheme

The three aforementioned voting schemes are biased in ranking the model hypotheses. The majority voting (MV) scheme tends to favor model candidates with many parts because model candidates with fewer parts are less likely to accumulate the same number of votes as model candidates with many parts. On the other hand, both Bayesian voting schemes (BVO and BVVR) tend to favor model candidates with fewer parts because the Bayesian approach will give higher posterior probabilities to model candidates with fewer parts than model candidates with many parts. BVVR is less biased than BVO, but a part with larger visual relevance has higher chance of being distorted when the object is slightly transformed, resulting in a lower rank in the correct model hypothesis. Thus, we need to design an unbiased voting scheme for ordering the model hypotheses regardless of the number of parts in the model candidates.

Let $S(M, p_i)$ represent the score for model M when p_i is observed in the scene

and $N(p_i)$ the total number of model hypotheses indexed by p_i. Let $\hat{P}(M|p_i)$ be the indexing power of p_i as defined in BVVR. Then $S(M, p_i)$ is computed as

$$S(M, p_i) = \begin{cases} [1.0 + \hat{P}(M|p_i)N(p_i)]\delta(p_i, M) & \text{if } \hat{P}(M|p_i) < 1.0 \\ 20.0 & \text{if } \hat{P}(M|p_i) = 1.0 \end{cases} \tag{4.9}$$

Note that $\hat{P}(M|p_i)$ is the indexing power of p_i for model M and when it is 1.0, it indicates that model M is present in the scene, thus a very large score is given reflecting this fact. Those model hypotheses with the indexing power $\hat{P}(M|p_i)$ less than the average $\frac{1}{N(p_i)}$ will have $\hat{P}(M|p_i)N(p_i)$ less than 1.0; in this case, MV outweighs BVVR. On the other hand, those model hypotheses with the indexing power greater than the average will have score higher than 1.0; then, BVVR outweighs MV. Thus, models with more parts will have less indexing power on their parts and $S(M, p_i)$ is scored mainly from MV, while models with less parts will have more indexing power on their parts and $S(M, p_i)$ is scored mainly from BVVR. In this way, the voting scheme balances the score between the models with various numbers of parts. Then the ranking of the model candidates is assigned by

$$R_{HYBRID}(M) = \sum_{i=i}^{m} S(M, p_i)$$

where $R_{HYBRID}(M)$ denotes the score of the model hypothesis M obtained via this hybrid voting scheme.

4.4 Experimental Results

To demonstrate the viability of the proposed indexing scheme for 3D object recognition, we intend to show the following: (1) the indexing scheme can provide a reasonable set of model hypotheses for verification/refutation; (2) model hypotheses are pruned significantly by the proposed voting schemes which are based on image evidence and prior model constraints; and (3) the system can recognize partially-occluded objects.

Experiments were conducted using a database of twenty 3D objects and eighty 2D objects with a total of 658 aspect models. Four sets of test data were used in simulations:

- Data Set ♯1: a copy of all model contour parts from the database (perfect data);

- Data Set ♯2: 2D distortion on Data Set ♯1 (2D distortion includes 2D transformation and Gaussian noise added to the contour);

- Data Set ♯3: 3D distortion on Data Set ♯1 (an extreme view of each model aspect was used) plus Gaussian noise 0 mean and 2 pixels standard deviation added to the contour;

- Data Set ♯4: 2D distortion on test data set ♯3 (i.e., 2D plus 3D distortion on perfect data which is similar to real data).

4.4.1 The Indexing Result

To evaluate the performance of our indexing scheme, we are interested in (1) how well the indexing scheme prunes model-aspect hypotheses and (2) how well the ordering of hypotheses limits the number of the verifications required. Four voting schemes, MV, BVO, BVVR, and HYBRID as described in Section 4.3.4, were used to study how they order the hypotheses. Table 4.1 shows some statistics obtained from these four voting schemes where the attribute tolerance criteria:$(e_\varsigma, e_\tau, e_\rho, e_{\phi_1}) = (10, 0.1, 10, 0.01)$ was used.

Table 4.1: Some statistics for Data Set ♯4 with 578 test objects.

Number of Parts	Frequency	Total Number of Hypotheses Generated	Voting Scheme's Total Rank			
			R_{MV}	R_{BVO}	R_{BVVR}	R_{HYBRID}
1	15	167	89	89	89	89
2	35	12154	1243	294	834	714
3	64	21568	3057	744	1171	1533
4	103	37099	4199	2419	1977	1975
5	121	45741	3528	2342	2272	2150
6	107	47473	1271	2855	1841	877
7	72	34398	1314	2815	1636	964
8	38	18234	727	2020	1803	826
9	21	10643	293	1104	741	491
11	2	947	2	98	27	2
Expected Value		395 (228424/578)	27	25	21	16

We can make the following observations from Table 4.1: (i) The number of expected aspect-model hypotheses indexed (395) is relatively large compared to the total number of aspect models (658) in the database; however, the expected rank, for example, 16 for R_{HYBRID}, is relatively small. This means than most of the aspect-model hypotheses can be ruled out by ordering the retrieved hypotheses. (ii) For

objects with more than five parts, MV outperforms (i.e., has lowest total rank) the Bayesian voting schemes, BVO and BVVR; for objects with less than five parts, BVVR and BVO outperform MV. This result verifies our hypothesis stated in the Section 4.3.4 that MV tends to favor models with more parts, while BVO and BVVR favors models with fewer parts. This phenomenon is also depicted in Figure 4.9. (iii) HYBRID outperforms the other three voting schemes.

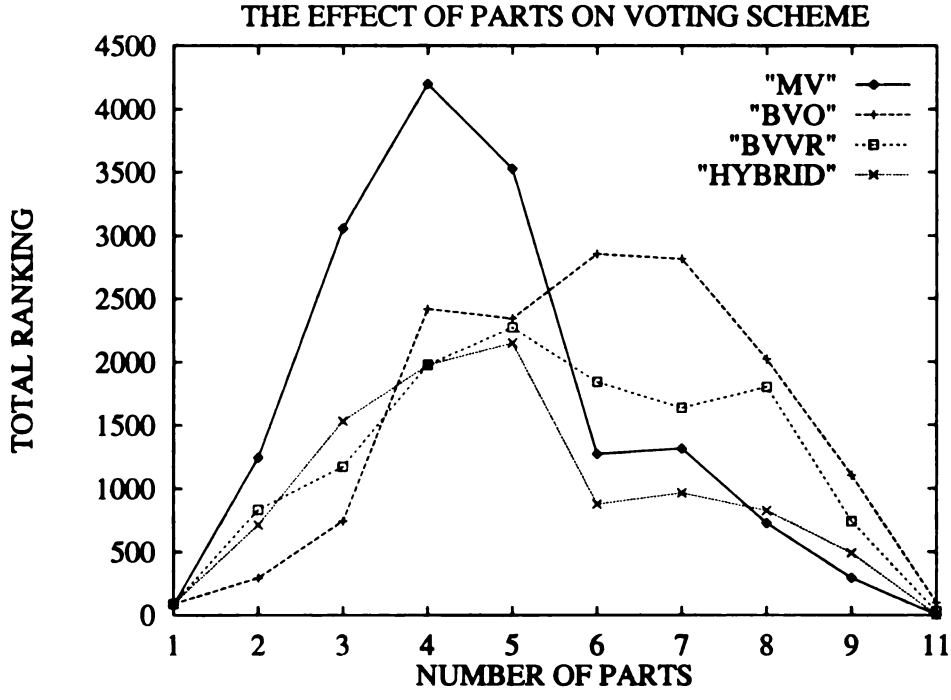


Figure 4.9: The effect of the number of object parts on voting schemes. Data Set ♯4 and the attribute tolerance criteria $(e_\varsigma, e_\tau, e_\rho, e_{\phi_1}) = (10, 1, 10, 0.01)$ were used.

Table 4.2 lists the number of average hypotheses generated (N_h), number of model aspects in the database (N_{db}), the reduction rate (R), defined as $(N_{db} - N_h)/N_{db})$, and the indexing percentile, defined as $(N_{db} - R_{HYBRID})/(N_{db} - 1)$. As indicated in Table 4.1, the indexing attributes do not provide a good discriminating mechanism (395 out 658 hypotheses were generated). This is because they are scale invariant.

If scale variant attributes, such as the *area* and *perimeter* of a part, in addition to scale invariant attributes, were used in indexing, the aspect-model hypotheses could be pruned significantly as shown in Table 4.2 where the reduction rate increases from 40% to 87.1% and the indexing percentile increases from 0.977 to 0.992. The significant reduction in the number of hypotheses passed to the verification procedure makes that indexing scheme more practical.

Table 4.2: The effect of using additional variant attributes in indexing.

Data Set ♯4 with 578 test objects was used Attribute tolerance criteria $(e_\varsigma, e_\tau, e_\rho, e_{\phi_1}) = (10, 1, 10, 0.01)$					
Indexing Scheme	N_{db}	N_h	R (%)	R_{HYBRID}	Indexing Percentile
Scale Invariant	658	395	40.0	16	0.977
Scale Variant & Invariant	658	85	87.1	6	0.992

To avoid those indexed hypotheses randomly clustered on aspect-models, we adopt the pose consistency grouping on the retrieved hypotheses. Table 4.3 shows the total ranking before and after the hypothesis grouping. For the HYBRID voting scheme, the total ranking drops from 13579 to 9621 with a pruning rate of 29.1%. This result indicates that the retrieved hypotheses should be grouped based on their pose consistency before the hypothesis ordering to improve the recognition performance.

A thorough study of the performance of the voting schemes on four sets of test data is given in Table 4.4. It can be seen from Table 4.4 that the tolerance on attributes plays an important role in indexing: when the tolerance is loose, the number of indexing failures decreases while the number of verification required, as indicated by the average rank, increases; and when the tolerance is tightened, the number of

Table 4.3: Hypothesis pruning via pose consistency grouping.

Data Set ♯4 with 578 test objects was used Attribute tolerance criteria $(e_\varsigma, e_\tau, e_\rho, e_{\phi_1}) = (10, 1, 10, 0.01)$				
Hypothesis Voting Scheme	MV	BVO	BVVR	HYBRID
Total Ranking Before Pose Consistency	18112	15939	14294	13579
Total Ranking after Pose Consistency	15723	14780	12391	9621
Hypothesis Pruning Rate (%)	13.2	7.3	13.3	29.1

verifications required decreases but the number of indexing failures increases. The choice of the attribute tolerance depends on the percentage of indexing failures (false dismissal) a system allows. Table 4.4 also shows that HYBRID outperforms the other three voting schemes in all four test data sets. A more detailed comparison is depicted in Figure 4.10. These results suggest that the HYBRID voting scheme should be used in ordering hypotheses to reduce the number of verifications required during the recognition. The execution time for the indexing scheme to generate hypotheses at run-time takes about 0.1 seconds on a SUN Sparc 20.

4.4.2 Recognition Results

We have shown the viability of the proposed indexing scheme for 3D object recognition through hypotheses grouping and ordering. The efficiency of the indexing scheme can also be achieved if tighter tolerances of indexing attributes are used but at the risk of more indexing failures, a situation which most recognition systems attempt to avoid. Sixty test images containing objects with sculptured surfaces (half of them containing partially occluded objects), were used to evaluate the performance of the

Table 4.4: Performance of hypothesis voting schemes.

ATC♯	Data Set ♯	Average Ranking				Indexing Failures
		R_{MV}	R_{BVO}	R_{BVVR}	R_{HYBRID}	
1	1	1.00	1.00	1.00	1.00	0
	2	1.06	1.05	1.04	1.06	31
	3	1.33	1.26	1.41	1.41	446
	4	1.41	1.32	1.45	1.44	458
5	1	1.12	1.13	1.14	1.12	0
	2	1.67	2.52	2.11	1.46	1
	3	17.61	14.47	11.93	10.56	25
	4	19.00	15.44	12.85	11.81	28
10	1	1.55	1.59	1.76	1.29	0
	2	3.46	4.80	5.43	2.89	0
	3	26.39	24.69	20.96	15.69	2
	4	27.20	25.57	21.44	16.65	2
15	1	2.29	2.33	2.74	1.50	0
	2	5.55	6.73	7.43	3.40	0
	3	32.11	30.75	25.66	19.99	0
	4	34.37	31.92	26.36	20.77	0
20	1	3.27	3.19	3.87	1.75	0
	2	6.87	7.91	8.86	3.48	0
	3	36.95	38.09	30.11	24.05	0
	4	37.94	40.42	31.12	24.16	0
ATC denotes attribute tolerance criteria $(e_\varsigma, e_\tau, e_\rho, e_{\phi_1})$ ATC♯n is defined as (n, 0.1n, n, 0.001n)						

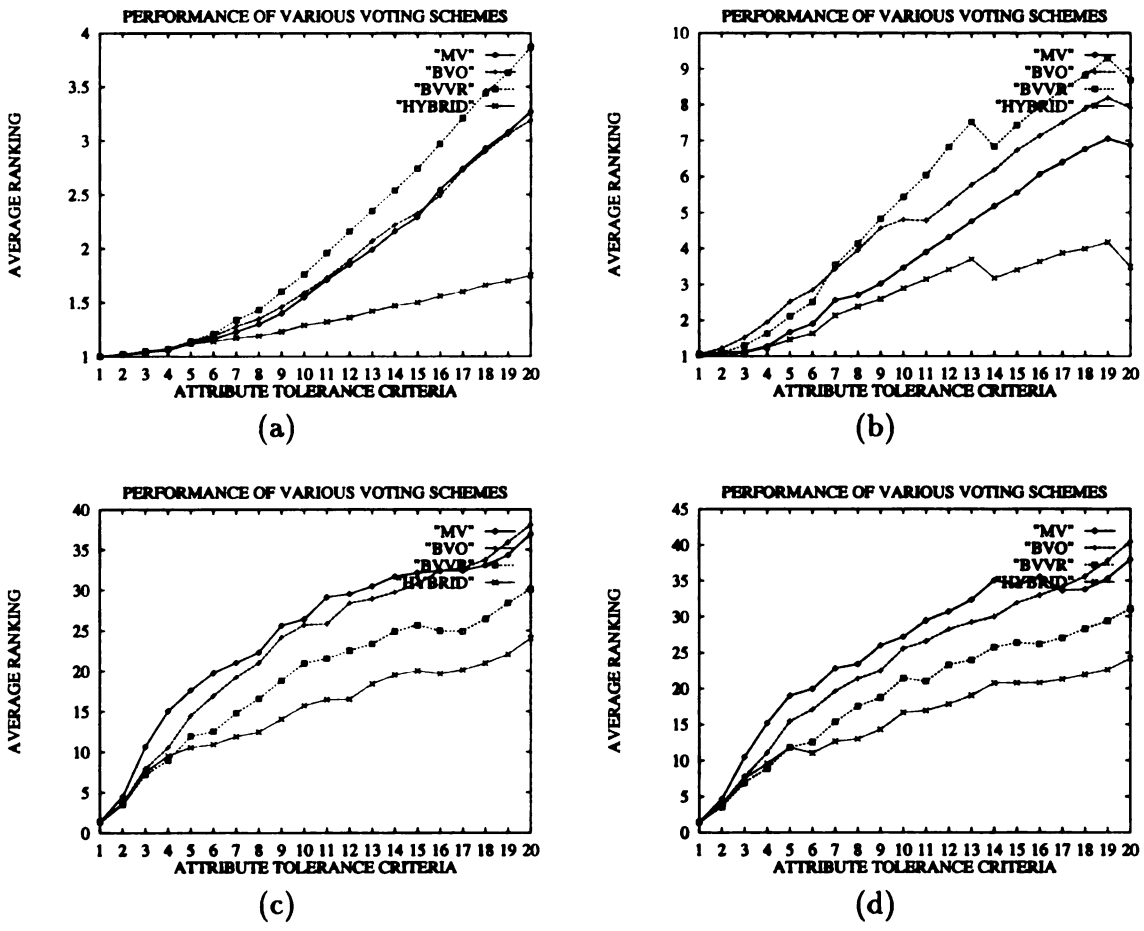
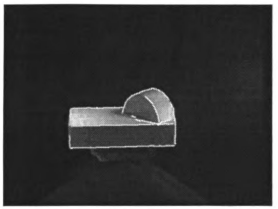


Figure 4.10: Comparison of the performance of hypothesis voting schemes. (a) Data Set ♯1 with 658 objects. (b) Data Set ♯2 with 658 objects. (c) Data Set ♯3 with 578 objects. (d) Data Set ♯4 with 578 objects.

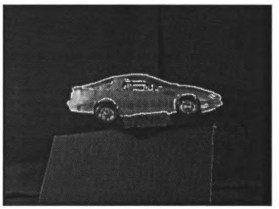
recognition system. Figures 4.11 and 4.12 show a few of recognition results. The rest of the results will be presented in the next chapter. Table 4.5 illustrates the performance of the recognition system. Figures 4.11 and 4.12 show the test images after indexing and recognition via alignment: the aligned model edgemaps are overlaid on the input images. Table 4.5 gives details on the indexing and alignment steps. It can be seen from Table 4.5 that the indexing step generates too many hypotheses (column 1) from most of the test images, but through hypotheses grouping and ordering, only a few hypotheses (column 2) need to be checked in the verification phase. For the test images *cup* and *soap*, they both are convex objects, as can be seen in Figure 4.11(c) and (d), and contain only one part. As a result, only one hypothesis was generated. However, the indexing would fail if they are partially occluded. The indexing failure issue will be addressed in Chapter 5. The reason for the low number of required verifications was that a tight attribute tolerance criteria was used to prune the incorrect hypotheses and that only a small portion of the parts corresponding to the correct model was occluded (see Figure 4.12). The final 2 columns of Table 4.5 illustrate that alignment was achieved within 1.4 pixels MSE on the edgemaps within 18 iterations of the hill-climbing procedure. Note that only the non-occluded contours of the test object were used to compute the MSE. The goodness of the fit is also shown in Table 4.5. In most cases, the pose parameters were achieved within reasonable ranges.

Table 4.5: Results of model indexing and hypothesis verification.

Number of models in the database: 658. Attribute tolerance criteria $(e_\varsigma, e_\tau, e_\rho, e_{\phi_1}) = (5.0, 1.0, 5.0, 0.005)$ Input parameters for alignment: $E_t = 1.6$ pixels and $I_{max} = 30$ iterations.									
Object	Model Indexing		Hypotheses Verification (Pose Estimation)						
	N_{HG}	Rank	E_{t_x}	E_{t_y}	E_s	E_α	$E_{\mathbf{N}}$	MSE	♯Iter
blockB1	102	1	1	0	0.001	0.7	0.03	0.83	4
camaro18	26	1	1	1	0.000	1.7	0.09	1.06	18
cup	1	1	1	1	0.006	0.2	0.20	1.05	12
soap	1	1	2	0	0.002	0.4	0.05	1.02	3
taurus1	113	1	0	2	0.005	0.3	0.28	1.31	5
truck4	217	1	3	2	0.012	0.5	0.04	0.95	8
elephant1†	160	1	2	0	0.010	1.0	0.06	0.97	10
gorilla1†	167	2	3	3	0.005	0.8	0.46	1.26	4
lion10†	141	2	2	1	0.005	0.5	0.24	1.37	6
phone13†	60	1	0	2	0.007	0.7	0.03	0.86	5
sprayer7†	35	1	1	0	0.010	1.7	0.33	0.98	3
swan2†	131	1	2	1	0.007	0.4	0.13	0.75	3
N_{HG} represents the number of hypotheses generated. $E_p = \|\hat{p} - p_{true}\|$ where $\hat{p}$ is estimated parameter value of p_{true}. The measure unit for E_{tx} & E_{ty} is pixels, for E_α is degrees, for $E_{\mathbf{N}}$ is radians. † indicates that the test object is partially occluded.									

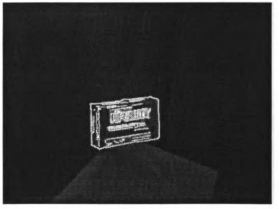


(a) blockB1

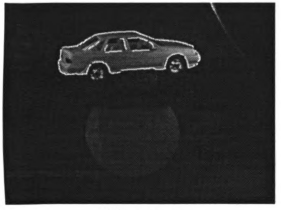


(b) camaro18

(c) cup



(d) soap

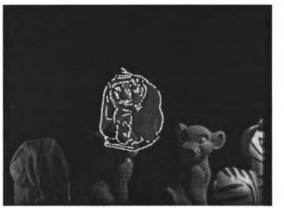


(e) taurus1

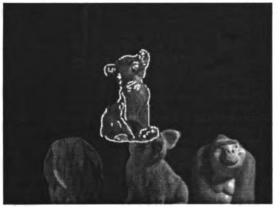
(f) truck4

Figure 4.11: Examples of hypothesis verification via model fitting.

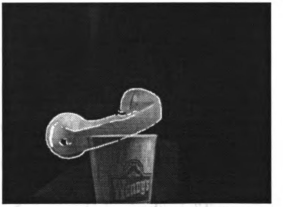
(a) elephant1



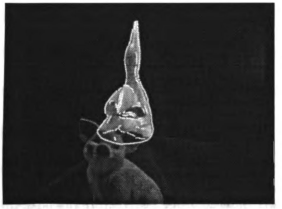
(b) gorilla1



(c) lion10



(d) phone13



(e) swan2

(f) sprayer7

Figure 4.12: More examples of hypothesis verification via model fitting.

4.5 Summary

We have proposed a new shape-based indexing scheme applicable to a large class of rigid curved objects. Indexing is based on invariant attributes extracted from parts on the object silhouette. We have presented a part representation that is invariant under 2D affine transformation (rotation, translation and scaling), and that provides robustness for effectively supporting indexing in the presence of occlusion and spurious noise. We have incorporated the pose consistency constraint to increase the accuracy of selecting model-aspect hypotheses. We have also investigated various hypothesis voting schemes that exploit prior knowledge to order model-aspect hypotheses for efficient search for candidate models. When combined into a complete system, these techniques make progress toward improving accuracy and efficiency by pruning false hypotheses and minimizing unnecessary verifications.

The work on indexing was framed within the general alignment paradigm proposed by David Lowe [76]. Our recognition experimental results have shown that we have made good progress toward a general alignment system where models can be taught via input imagery. This system should work well on some existing recognition problems. Although we have only used shape features, the paradigm can easily accommodate other types, such as color or texture features. Moreover, we could also include shape features derived from the internal object edges. In our experiments, we have made stronger assumptions on the quality of object contours available from the input image than were made by Lowe. The part theory applies, however, to open contours so it is not required that an entire object is segmented from the background.

Surely, degraded input implies increased ambiguity and computational load. We will study recognition in more difficult imagery in future work.

Chapter 5

Object Recognition and Tracking

This chapter is concerned with the integrated recognition system as well as the applicability of this system for object tracking. We have described the object representation in Chapter 2, the matching algorithm in Chapter 3, and the indexing techniques in Chapter 4, In this chapter, we propose a hierarchical verification strategy to cut down the recognition time and explain how we can track a moving object in an image sequence.

The important feature of the integrated recognition system is that it does not need to verify all the false model hypotheses at a full length. False model hypotheses can be repudiated at an early stage of the verification process. Only hypotheses of correct models need to be verified at a full length. Thus, we propose conservative threshold values to determine when a false model hypothesis should be rejected during the verification process.

This chapter is divided into 3 sections. Section 5.1 proposes a verification strategy to expedite the recognition. Section 5.2 demonstrates the applicability of the

recognition system for tracking moving objects in image sequences. A summary of the experiments is given in Section 5.3.

5.1 Object Recognition

The object recognition system operates within the alignment paradigm. Four stages are involved in this framework: (1) model building, (2) feature extraction, (3) model-aspect hypothesis generation via indexing, and (4) model-pose hypothesis verification through aligning the 3D projected model with the object image features. Both model building and index construction are processed off-line. The computation time for feature extraction is fixed. Thus, the recognition time is primarily determined by the efficiency of hypothesis generation and verification. The indexing scheme must compensate for noise and view variation by increasing the indexing attribute tolerance, thus, increasing the number of incorrect model-aspect hypotheses that are generated. To minimize the recognition time, the indexing scheme exploits some prior knowledge to reduce the number of verifications required during recognition. However, the recognition system may still need to repudiate a few false model-pose hypotheses before the correct one is verified. For example, the experimental results in Section 4.4 indicate that a significant percentage of model-aspect hypotheses can be removed from consideration via hypothesis ordering, but for a loose indexing attribute tolerance, such as ATC♯5 in Table 4.4, on the average, about 10 model-pose hypotheses still need to be tried before the correct one is verified. This slows down the recognition significantly. Consequently, shortening the verification time for false model-pose

hypotheses becomes a necessity for the recognition system to be more efficient. As described next, we incorporate a hierarchical verification strategy into the recognition system to expedite the recognition.

5.1.1 Hierarchical Verification

The model-pose hypothesis verification is carried out through robust pose estimation, as described in Chapter 3, which uses Newton's method with Levenberg-Marquardt minimization (N-L-M). The experimental results in Chapter 3 show that for correct model aspects with good initial parameter estimates relatively close to true parameter estimates, the N-L-M method converges within a few iterations. But for wrong model aspects, the N-L-M method tends to wander around the parameter space, resulting in a slow convergence to a local minimum or no convergence at all. Accordingly, a tremendous amount of computation time is wasted on verifying incorrect model-pose hypotheses. Thus, it is essential to reduce the number of iterations in the N-L-M method when incorrect model-pose hypotheses are verified.

The edgemap matching algorithm (i.e., the N-L-M method), presented in Figure 3.2 of Chapter 3, uses two parameters: the maximum number of iterations allowed, I_{max}, and the matching error threshold, E_t, to determine when the algorithm should terminate. For a correct model aspect, a large value of I_{max} and a small value of E_t tend to lead the algorithm to either converge with a mean-squared distance error (MSE) less than E_t or to select the best match after I_{max} iterations are tried. In either case, the verification is thoroughly carried out, and the result can be used

to determine whether to accept or refute a model-pose hypothesis. However, such an exhaustive verification for an incorrect model-pose hypothesis is neither necessary nor possible—the incorrect model-pose hypothesis can be detected after a few iterations of the N-L-M method.

To expedite the recognition, we propose a hierarchical verification strategy as follows. Three situations may occur during the verification process:

(1) fitting the correct model aspect to the object image features;

(2) fitting the wrong model to the object image features;

(3) fitting the correct model but with wrong aspect to the object image features.

Normally, for those model-pose hypotheses where the model aspect and the observed object stem from the same object model but different aspects, the fitting error indicated by the MSE is in between that from situation (1) and that from situation (2). We can vary the matching error threshold, E_t, and the maximum number of iterations allowed, I_{max}, to control the thoroughness of a verification. Three levels of verification, *coarse, middle*, and *fine*, are defined based on the various values of I_{max} and E_t. During the recognition, model-pose hypotheses are verified in the order of their rank obtained from indexing. Each ranked model-pose hypothesis is first verified at the coarse level; if it does not survive at the coarse level, it is removed from further verification; if it survives, then it is verified at the middle level; if it survives again at this level, then it continues to be verified at the fine level; otherwise, it is rejected. The specified error threshold of the fine level will be used to accept a model-pose hypothesis whenever the fitting error is below this error threshold during

Input: a set of ranked model-pose hypotheses generated via indexing; three sets of threshold values, (I^c_{max}, E^c_t), (I^m_{max}, E^m_t), and (I^f_{max}, E^f_t) for coarse, middle, and fine level of verification, respectively.

Outputs: either failure or the model identity and pose parameters $\vec{\omega}$.

Processing: this verification strategy uses the N-L-M method to compute the pose parameters $\vec{\omega}$ and the fitting error MSE iteratively for each hypothesis until the correct one is found; otherwise report failure. The iterations repeat the following steps:

(1) Get the next lower ranked model-pose hypothesis;

(2) Initialize the iteration counter: $Iter = 1$;

(3) If $MSE \leq E^f_t$, then accept the current hypothesis (i.e., report the pose parameter $\vec{\omega}$ and the model identity) and terminate the whole verification process;

(4) If $MSE > E^c_t$ and $Iter > I^c_{max}$, then reject the current hypotheses and go to step (1); otherwise go to the next step;

(5) If $MSE > E^m_t$ and $Iter > I^m_{max}$, then reject the current hypotheses and go to step (1); otherwise go to the next step;

(6) If $MSE > E^f_t$ and $Iter > I^f_{max}$, then reject the current hypotheses and go to step (1); otherwise go to the next step;

(7) Increase $Iter$ by 1 and go to step (3).

Figure 5.1: The hierarchical verification strategy.

verification. Once a ranked model-pose hypothesis is accepted, the whole verification process terminates; otherwise this hierarchical verification process continues with the next lower ranked model-pose hypothesis. The hierarchical verification strategy is sketched in Figure 5.1.

5.1.2 Monte Carlo Experiments

In this section, we describe several Monte Carlo experiments on a large number of test images conducted to study what empirical values for E_t and I_{max} should be used for verifications at the coarse, middle, and fine level, respectively.

Fitting Correct Model Aspects to Images

In this experiment, six randomly chosen views for each of 16 models and one view for each of 4 single model aspects in our database were generated as the test images. Thus, we had 100 randomly selected test images for 100 (out of 576) model aspects. These 100 model-pose hypotheses were tested via the N-L-M method.

Table 5.1 shows the results, which were collected after fitting 100 model aspects to their corresponding test images. The cell (I_{max}, E_t) in Table 5.1 indicates the number of model-pose hypotheses in which the N-L-M method converges within I_{max} iterations and with an MSE less than E_t. For example, for $I_{max} = 5$ iterations and $E_t = 4.0$ pixels, 98 (out of 100) model-pose hypotheses are accepted (i.e., survive); two model-pose hypotheses did not survive because either bad initial parameter estimates or small values of I_{max} were used in the N-L-M method. We used another set of initial parameter estimates for these two model-pose hypotheses, and in both cases,

Table 5.1: Number of verifications in fitting correct model aspects to images as a function of E_t and I_{max}.

100 model-pose hypotheses were used.								
I_{max} (iterations)	Error Threshold: E_t (pixels)							
	1.0	1.5	2.0	2.5	3.0	4.0	5.0	6.0
3	26	46	63	80	89	96	97	99
4	39	60	72	87	95	97	99	99
5	45	63	81	91	95	98	99	99
6	47	66	85	93	96	98	99	100
7	48	68	88	93	97	99	99	100
8	51	70	89	95	97	99	99	100
9	51	74	91	96	97	99	99	100
10	52	75	92	97	98	99	100	100
11	52	75	93	97	99	100	100	100
12	53	76	94	98	100	100	100	100

the N-L-M method converged within 5 iterations and with a fitting error less than 2.0 pixels MSE. The N-L-M method also converged within 15 iterations when the original initial parameter estimates were used. This suggests that for verification at the fine level, the N-L-M method should use a few sets of initial parameter estimates and a large value for I_{max} to ensure that correct model-pose hypotheses are always accepted. It can be seen from Table 5.1 that with $I_{max} = 6$ iterations and $E_t = 6.0$ pixels, all 100 model-pose hypotheses survive. Thus, for this set of model aspects, $I_{max} = 6$ iterations and $E_t = 6.0$ pixels are good criteria for verification at the coarse level to refute wrong model-pose hypotheses.

Fitting Incorrect Model Aspects to Images

In this experiment, we hoped to demonstrate that most wrong model-pose hypotheses are refuted through verification at the coarse level. For each of the aforementioned

Table 5.2: Number of verifications in fitting wrong models to images as a function of E_t and I_{max}.

1000 model-pose hypotheses were used.							
I_{max} (iterations)	Error Threshold: E_t (pixels)						
	1.0	2.0	2.5	3.0	4.0	5.0	6.0
1	0	0	0	0	0	1	3
2	0	0	0	1	1	3	4
3	0	0	1	1	2	3	7
4	0	0	1	1	3	3	7
5	0	0	1	1	3	3	9
6	0	0	1	1	3	4	11

100 test images, 10 (excluding the correct one) randomly selected model aspects were used to form 10 model-pose hypotheses. All 1000 model-pose hypotheses were tested using the N-L-M method.

Table 5.2 shows some results from this experiment. As can be seen in Table 5.2, with $E_t = 6.0$ pixels and $I_{max} = 6$ iterations (i.e., the criteria for verification at the coarse level), only 11 out of 1000 model-pose hypotheses survived the coarse level. Table 5.3 lists the fitting errors of these 11 model-pose hypotheses. It shows in Table 5.3 that the best 3 model-pose hypotheses are either with the same object model (*sprayer7-sprayer10*) or with neighboring aspects (*zebra1-zebra20* and *phone7-phone6*). This suggests that the goal for verification at the middle level is to refute those model-pose hypotheses in which the model aspect and the observed object stem from the same object model but different aspects, or from similar object models.

Table 5.3: Incorrect model-pose hypotheses that survived at the coarse level.

Model Aspect	Test Image	MSE (pixels)
sprayer7	sprayer10	2.3
zebra1	zebra20	3.7
phone7	phone6	3.8
blockB7	soap	4.9
camaro20	taurus2	5.4
lion29	lion10	5.4
elephant1	gorilla16	5.5
soap	taurus1	5.8
gorilla1	gorilla16	5.8
soap	taurus5	5.9
soap	zebra20	5.9

Fitting Models to Neighboring Aspects

To study the effect of neighboring aspects on the fitting error, 100 randomly selected model-pose hypotheses were generated, in which model aspects and their corresponding test images were from neighboring aspects. Table 5.4 lists the results of these 100 model-pose hypotheses. As shown in Table 5.4, only 5 out of 100 model-pose hypotheses survived when I_{max} and E_t were 12 iterations and 3.0 pixels, respectively. Note that with $I_{max} = 12$ iteration and $E_t = 3.0$ pixels, all correct model-pose hypotheses survive as shown in Table 5.1. This suggests that for verification at the middle level, $I_{max} = 12$ iterations and $E_t = 3.0$ pixels are good criteria to remove model-pose hypotheses in which model aspects and test images are from the same object model but different aspects.

From the results of above three experiments, we are able to determine the empirical values for I_{max} and E_t to be used in verifications at various levels. These values are shown in Table 5.5. Note that two sets of initial parameter estimates are used for

Table 5.4: Number of verifications in fitting models to their neighboring aspects as a function of E_t and I_{max}.

100 model-pose hypotheses were used.							
I_{max} (iterations)	Error Threshold: E_t (pixels)						
	1.0	2.0	2.5	3.0	4.0	5.0	6.0
3	0	0	0	3	13	23	40
4	0	0	0	3	17	27	45
5	0	0	2	4	20	28	50
6	0	0	2	4	20	32	51
7	0	0	2	4	20	36	52
8	0	0	3	4	20	39	56
9	0	0	3	4	21	43	57
10	0	0	3	5	24	44	59
11	0	0	3	5	25	45	61
12	0	0	3	5	26	47	63

verification at the fine level to prevent the N-L-M method from getting stuck in local minima. Normally, one set of initial parameter estimates is sufficient for the N-L-M method to converge because the basin of convergence is broad as shown from the experimental results in Chapter 3. The extra set of initial parameter estimate is used to further guarantee the convergence of the N-L-M method. Thus, if a model-pose hypothesis, using one set of initial parameter estimates in the N-L-M method, passes the verification at the middle level, but fails at the fine level, then the other set of initial parameter estimates is used for the N-L-M method to test this model-pose hypothesis again. If one set of initial parameter estimates causes the N-L-M method to converge, the other set will be ignored.

Table 5.5: Parameters for verifications at various levels.

Level	I_{max} (iterations)	E_t (pixels)
coarse	6	6.0
middle	12	3.0
fine	18	2.0

An Example

We provide an example to demonstrate how this hierarchical verification strategy expedites the recognition. We use the test image for model aspect *gorilla20* to index into the database. 137 model-aspect hypotheses are retrieved. The correct model aspect, *gorilla20*, is ranked as the 5th among these 137 model-aspect hypotheses. Table 5.6 lists the number of iterations required for the first five model-pose hypotheses using the hierarchical verification strategy. The input parameters for verifications at various levels as shown in Table 5.5 were used. Since model aspect *gorilla3* is neighboring to model aspect *gorilla20*, the corresponding model-pose hypothesis passes the verification at the coarse and middle levels, respectively; the rejection at the fine level costs totally 18 iterations in the N-L-M method. Model aspect *gorilla13* is from the same object model with model aspect *gorilla20*; the corresponding model-pose hypothesis only passes the verification at the coarse level; the rejection at the middle level costs 12 iterations. Both model-pose hypotheses for *lion31* and *sharpener5* are rejected at the coarse level, and each rejection costs 6 iterations. When the correct model aspect *gorilla20* was used in alignment, the N-L-M method converged with an MSE less than the specified error threshold E_t of the fine level; the N-L-M method accepted the hypothesized model aspect and pose, and any further testing was ter-

minated. The total cost for the hierarchical verification was 47 iterations as shown in Table 5.6. The total cost for the non-hierarchical verification would be 77 (i.e., $18 \times 4 + 5$) iterations; each wrong model-pose hypothesis costs 18 iterations, and the correct one costs only 5 iterations. Let I_N and I_H denote the number of iterations required for the non-hierarchical and hierarchical verification, respectively. Define the speedup of using the hierarchical verification over the non-hierarchical verification as $S = (I_N - I_H)/I_H$. Then the speedup in this example is 63.8%. It is straightforward to derive the upper and lower bounds of S. Suppose the rank for the correct aspect-model hypothesis is N, and the corresponding model-pose hypothesis costs m iterations for verification. The upper bound of S occurs when the N-1 incorrect model-pose hypotheses, ranked from 1 to N-1, are tested and rejected at the coarse level. Then $S = \frac{[18\times(N-1)+m]-[6\times(N-1)+m]}{6(N-1)+m}$. Thus, when N is very large, the upper bound of S is approximately 200%. The lower bound of S occurs when $N = 1$, i.e., there is no wrong model-pose hypothesis to reject; in this case $S = 0$. Note that the hierarchical verification strategy would gain some speedup only if it can reject wrong model-pose hypotheses at the early stage of the N-L-M method. More examples are given in the next section.

5.1.3 Recognition Results

In this section, we intend to show the efficiency and viability of the proposed recognition system. The hierarchical verification strategy is incorporated in the recognition system to improve the efficiency of the recognition. The viability of the recognition

Table 5.6: Hierarchical verification for model aspect: gorilla20.

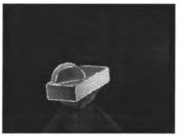
Model Indexing		Hypothesis Verification			
Rank	Model Aspect	Coarse	Middle	Fine	Iterations
1	gorilla3	pass	pass	reject	18
2	gorilla13	pass	reject		12
3	lion31	reject			6
4	sharpener5	reject			6
5	gorilla20	accept			5

system is demonstrated through alignment results of sixty real sculptured objects; 28 of them are with non-occluded objects, and 32 of them are with partially occluded objects. Tables 5.7 and 5.8 give details on the indexing and alignment steps. Figures 3.10, 3.11, 4.11, 4.12, 5.2, 5.3, and 5.4 show the test images after indexing and recognition via alignment: the aligned model edgemaps are overlaid on the input images.

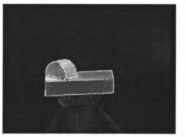
Recognition of Non-occluded Objects

Table 5.7 lists the number of model-aspect hypotheses generated (N_H) via indexing, the rank of the correct model aspect (R_k), the goodness of the fit ($E_{tx}, E_{ty}, E_s, E_\alpha, E_{\vec{N}}$), the fitting error (MSE), the number of iterations required for the hierarchical verification (I_H), the number of iterations required for the non-hierarchical verification (I_N), and the speedup of using the hierarchical verification (S), defined as $(I_N - I_H)/I_H$. 3 (out of 28) test images contain objects with only one part in their silhouettes; these three objects are *cup* and *soap* in Figure 4.11(c) and (d), respectively, and *blockB10* in Figure 5.2(c). As shown in the column N_H

(a) blockB4

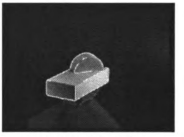


(b) blockB7



(c) blockB10

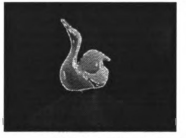
(d) blockB13



(e) blockB16

(f) phone1

(g) phone7



(h) swan13

(i) taurus2

(j) taurus3

(k) taurus4

(l) taurus5

Figure 5.2: Examples of recognition.

(a) taurus8

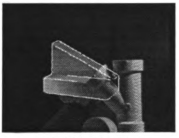
(b) taurus13

(c) taurus14

(d) taurus19

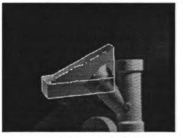
(e) taurus20

(f) blockA5



(g) blockA9

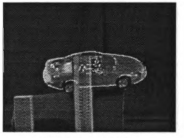
(h) blockA14



(i) blockA16

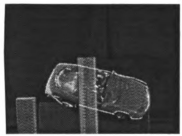
(j) camaro20

(k) camaro27



(l) camaro36

Figure 5.3: Examples of recognition.



(a) camaro37

(b) elephant20

(c) gorilla20

(d) mug7

(e) phone10

(f) sharpener1

(g) sprayer1

(h) sprayer25

(i) squirrel19

(j) squirrel25

(k) swan3

(l) zebra10

Figure 5.4: Examples of recognition.

of Table 5.7, both *soap* and *cup* have only 1 model-aspect hypothesis generated via indexing, but *blockB10* has 10. Since *blockB10* has only one part, all the 10 generated model-aspect hypotheses for *blockB10* will have the same scores weighted by any of the hypothesis ordering schemes as presented in Chapter 4; thus, the correct one is ranked as the 5th, i.e., the average rank. The rest of the test images contain objects with several parts in their silhouettes, and consequently, the number of model-aspect hypotheses generated from indexing increases. However, the HYBRID hypothesis ordering scheme as presented in Chapter 4 has effectively ranked the correct model-aspect hypothesis. As can be seen from the column R_k of Table 5.7, 27 (out of 28) test objects are ranked as the first, which implies only one verification. This indicates that the HYBRID hypothesis ordering scheme has effectively pruned the incorrect model-aspect hypotheses using the prior knowledge about the pre-stored models and the visual evidence of the observed objects. As indicated in the column S of Table 5.7, the integrated recognition system does not gain any speedup by using the hierarchical verification strategy because most of the correct model-aspect hypotheses require only one verification; however, the overall recognition time for each of these 27 objects, indicated by the column I_H of Table 5.7, is smaller than the recognition time for the test object, *blockB10*, which has gained some speedup ($S = 128\%$). Thus, the overall speed of the recognition system is not evaluated by the speedup S but by the total iterations required for verification.

The columns MSE and I_H of Table 5.7 illustrate that alignment was achieved within 2.0 pixels MSE on the edgemaps and within 28 iterations of the N-L-M method. The goodness of the fit is also shown in Table 5.7. In most cases, the pose parameters

Table 5.7: Recognition results for 28 non-occluded objects.

Number of model aspects in the database: 658. Indexing attribute tolerance criteria $(e_\varsigma, e_\tau, e_\rho, e_{\phi_1}) = (5.0, 1.0, 5.0, 0.005)$ Parameters for hierarchical verification are specified in Table 5.5.											
Object	Indexing		Hypothesis Verification (Pose Estimation)								
	N_H	R_k	E_{t_x}	E_{t_y}	E_s	E_α	$E_{\vec{N}}$	MSE	I_H	I_N	S(%)
blockB1	102	1	1	0	0.001	0.7	0.03	0.83	4	4	0
blockB4	114	1	2	0	0.006	0.1	0.33	0.77	5	5	0
blockB7	91	1	2	1	0.001	0.6	0.04	0.90	4	4	0
blockB10	10	5	3	0	0.004	1.5	0.12	1.04	28	64	128
blockB13	116	1	3	0	0.005	0.7	0.06	1.02	4	4	0
blockB16	111	1	1	0	0.005	0.5	0.21	0.96	4	4	0
camaro8	223	1	3	2	0.009	0.4	0.06	1.16	9	9	0
camaro18	26	1	1	1	0.004	0.4	0.03	1.06	16	16	0
cup	1	1	1	1	0.006	0.2	0.20	1.05	12	12	0
face*	113	1	0	0	0.002	2.2	0.20	1.39	7	7	0
phone1	50	1	0	0	0.005	0.7	0.08	1.24	4	4	0
phone6	71	1	1	0	0.000	0.4	0.06	1.08	6	6	0
phone7	82	1	1	0	0.002	1.0	0.08	1.28	5	5	0
sharpener10	88	1	3	0	0.001	2.4	0.13	1.36	8	8	0
soap	1	1	2	0	0.002	0.4	0.05	1.02	3	3	0
swan11	64	1	0	0	0.001	1.0	0.15	0.99	4	4	0
swan13	42	1	2	0	0.002	1.7	0.08	0.97	6	6	0
taurus1	113	1	0	2	0.005	0.3	0.28	1.31	5	5	0
taurus2	173	1	3	0	0.002	2.0	0.06	1.47	3	3	0
taurus3	153	1	3	0	0.002	2.3	0.04	1.65	3	3	0
taurus4	47	1	4	0	0.008	1.2	0.25	1.20	5	5	0
taurus5	58	1	2	0	0.004	0.8	0.08	0.89	3	3	0
taurus8	61	1	2	0	0.002	2.1	0.04	1.22	6	6	0
taurus13	30	1	3	1	0.000	1.1	0.21	1.38	6	6	0
taurus14	69	1	3	0	0.007	0.9	0.11	1.16	5	5	0
taurus19	147	1	8	1	0.018	4.9	0.39	1.58	18	18	0
taurus20	82	1	1	1	0.005	1.6	0.12	1.92	4	4	0
truck4	217	1	3	2	0.012	0.5	0.04	0.95	8	8	0
$(t_x, t_y, s, \alpha, \vec{N})$ represents the object pose where (t_x, t_y) is 2D translation, s the scale factor, and α and $\vec{N}$ the rotation angle and axis, respectively. E_p denotes $\|\hat{p} - p_{true}\|$ where $\hat{p}$ is the estimated value of the ground truth p_{true}. * ground truth not actually known (only approximation).											

were achieved within reasonable ranges. Note that for the test object *taurus19*, the large error in t_x (8 pixels off the ground truth) was due to the large error in s (1.8% off the ground truth). The error in t_x was compensated by the error in s, resulting in a reasonable fitting error (1.58 pixels MSE) on the edgemaps. Nevertheless, the convergence of the fitting was very slow (18 iterations), and also the errors in rotational parameters were quite large; 4.9 degrees off the ground truth for the rotation angle α, and 0.39 radians off the ground truth for the rotation axis $\vec{N}$. This provides an example in which an acceptable fitting error does not necessarily guarantee the accuracy of pose parameters. As discussed in Chapter 3, the accuracy of pose parameters is affected by the shapes of the objects. The execution time for each iteration of the N-L-M method, excluding I/O, is about 0.1 second on a Sparc 20, and the execution time for the indexing scheme to generate hypotheses for each test image, again excluding the time on loading the database into memory, takes about 0.1 second.

Recognition of Occluded Objects

Results with occluded objects are given in Table 5.8: the notations are the same as those in Table 5.7. Note that in the column R_k of Table 5.8, five test objects, *blockA1, blockA5, blockA9, blockA14*, and *blockA16*, have the symbol $-$, which denotes indexing failure. Because these five objects have only one part in their silhouettes and they are partially occluded, indexing fails. This is one break point of the indexing scheme. Without the help from indexing to prune the model-aspect hypotheses, a lot of verifications need to be performed. For example, for the test object *blockA1*, 34 model-pose hypotheses needed to be tried first because they were generated from

Table 5.8: Recognition results for 32 occluded objects.

Number of model aspects in the database: 658. Indexing attribute tolerance criteria $(e_\varsigma, e_\tau, e_\rho, e_{\phi_1}) = (5.0, 1.0, 5.0, 0.005)$ Parameters for hierarchical verification are specified in Table 5.5.											
Object	Indexing		Hypothesis Verification (Pose Estimation)								
	N_H	R_k	E_{t_x}	E_{t_y}	E_s	E_α	$E_{\vec{N}}$	MSE	I_H	I_N	S(%)
blockA1	34	-	0	2	0.007	0.4	0.16	1.02	2081	6233	199
blockA5	42	-	5	3	0.015	1.6	0.13	0.88	4200	6305	199
blockA9	44	-	1	2	0.010	1.2	0.43	1.09	4212	6321	200
blockA14	8	-	4	1	0.001	2.2	0.60	1.03	2000	5996	200
blockA16	44	-	0	0	0.003	0.1	0.33	1.29	4210	6319	199
camaro20	126	1	1	0	0.001	0.7	0.03	0.83	4	4	0
camaro27	25	1	2	0	0.009	0.2	0.14	1.79	4	4	0
camaro36	189	1	5	1	0.017	1.8	0.61	1.62	3	3	0
camaro37	170	1	1	2	0.005	1.2	0.03	1.74	2	2	0
can	181	1	0	0	0.003	0.8	0.05	0.94	4	4	0
elephant1	160	1	2	0	0.010	1.0	0.06	0.97	10	10	0
elephant20	239	25	4	5	0.006	2.0	0.16	1.06	179	437	144
gorilla1	167	2	3	3	0.005	0.8	0.46	1.26	10	22	120
gorilla20	137	5	3	0	0.009	0.5	0.01	0.85	47	77	64
lion10	141	2	2	1	0.005	0.5	0.24	1.37	12	24	100
lion29	240	12	1	0	0.000	0.5	0.06	1.56	77	203	164
mug7	194	59	1	1	0.008	3.9	0.80	1.02	388	1054	173
mug26	216	33	2	0	0.010	0.2	0.10	0.98	223	583	161
phone10	103	32	1	0	0.000	0.0	0.15	1.78	196	568	190
phone13	60	1	0	2	0.007	0.7	0.03	0.86	5	5	0
pig1	204	1	0	0	0.000	0.1	0.03	1.22	5	5	0
sharpener1	226	1	3	3	0.005	3.1	0.02	0.97	3	3	0
sprayer1	30	1	3	1	0.013	0.2	0.15	1.17	10	10	0
sprayer7	35	1	1	0	0.010	1.7	0.33	0.98	3	3	0
sprayer25	28	1	1	0	0.002	0.1	0.10	1.73	11	11	0
squirrel19	218	1	2	2	0.010	0.3	0.12	1.61	5	5	0
squirrel25	181	107	1	1	0.020	2.9	0.14	1.75	701	1913	173
swan2	131	1	2	1	0.007	0.4	0.13	0.75	3	3	0
swan3	146	1	3	1	0.008	0.1	0.12	0.93	10	10	0
truck10	78	1	2	2	0.009	0.7	0.07	0.95	10	10	0
zebra1	222	9	3	3	0.010	0.3	0.05	0.76	68	152	124
zebra20	265	5	5	5	0.009	1.1	0.01	1.19	27	75	178
$(t_x, t_y, s, \alpha, \vec{N})$ represents the object pose where (t_x, t_y) is 2D translation, s the scale factor, and α and $\vec{N}$ the rotation angle and axis, respectively. E_p denotes $\|\hat{p} - p_{true}\|$ where $\hat{p}$ is the estimated value of the ground truth p_{true}.											

indexing, but none of them would be accepted; then the rest of the model aspects are tested in some random order; on the average 317, i.e., (658-34)/2, model-pose hypotheses are verified, resulting in a tremendous number of verification attempts. The speedup gained by using the hierarchical verification strategy almost reaches the upper bound, i.e., $S = 200\%$. In this case, indexing does not help the system to alleviate the burden of verification; on the contrary, it increases the recognition time by requiring the system to perform an extra $N_H/2$ tests (were the indexing not provided, the system only needed to verify 329 model-pose hypotheses on the average). As can be seen in the column S of Table 5.8, the recognition system always gained some speedup whenever the correct model-aspect hypothesis is not ranked as the first. The column MSE of Table 5.8 illustrates that alignment was achieved within 2.0 pixels MSE on the edgemaps. Due to the indexing failures, the recognition system tested a lot of false model-pose hypotheses, resulting in a huge number of iterations of the N-L-M method, as indicated by the column I_H in Table 5.8. The goodness of the fit is also shown in Table 5.8. In most cases, the pose parameters were achieved within reasonable ranges, although in general not as good as the pose parameters for non-occluded objects as shown in Table 5.7.

The column R_k in Table 5.8 shows that the performance of the hypothesis ordering scheme deteriorated when the observed object was partially occluded. However, as long as the percentage of visual contour of non-occluded object parts is significant, the hypothesis ordering scheme still could effectively rank the correct model-aspect hypothesis; for 14 (out of 32) test objects, the correct model aspect ranked first. Table 5.9 lists the number of non-occluded *object* parts (N_{NP}), the number of *scene*

Table 5.9: The effect of visual contour on indexing.

Object	N_{NP}	N_{SP}	VC (%)	N_H	R_k
elephant20	1	9	28.8	239	25
gorilla1	2	11	25.7	167	2
gorilla20	1	7	19.7	137	5
lion10	3	8	24.1	141	2
lion29	2	9	15.3	240	12
mug7	1	9	13.3	194	59
mug26	1	10	13.6	216	33
phone10	1	5	22.8	103	32
squirrel25	1	11	17.2	181	107
zebra1	2	8	25.9	222	9
zebra20	3	10	23.2	265	5

parts (N_{SP}), the percentage of the visual contour of non-occluded object parts in the whole scene contour (VC), the number of model-aspect hypotheses generated via indexing (N_H), and the rank of the correct model-aspect hypothesis (R_k). As can be seen in Table 5.9, the reason for the high number of required tests for the test objects *elephant20, mug7, mug26, phone10*, and *squirrel25* is that these objects are partially occluded and only a small portion of the parts corresponds to the correct model aspect; for the test images *mug7*, only one part, i.e., the handle of the mug, (out of 9 scene parts) is able to index to the correct model aspect and its visual contour is relatively small (13.3%) as compared to the whole visual contour in the scene (see Figure 5.4 (d)). The number of hypotheses generated depends on the number of scene parts and the rarity of each scene part in the database. Normally, more scene parts will generate more hypotheses. The rank of the correct model aspect depends on the saliency of non-occluded parts in the object and the percentage of visual contour of

these parts in the scene. Thus, there is no particular regularity in Table 5.9, but, in general, a smaller percentage of visual contour of non-occluded object parts means a lower rank of the correct model-aspect hypothesis.

Recognition Failure

Several examples are given here to demonstrate two types of recognition errors, namely *false alarm* and *false dismissal.* Figure 5.5 shows the test images containing objects not existing in the model database. Table 5.10 lists the number of hypotheses generated via indexing, the number of hypotheses accepted, and the number of hypotheses rejected. For vehicles shown in Figure 5.5(a)-(d), edge contours (shown as

Table 5.10: Recognition results of objects not in the database.

Indexing attribute tolerance criteria $(e_\varsigma, e_\tau, e_\rho, e_{\phi_1}) = (4.0, 0.4, 4.0, 0.005)$ Parameters for hierarchical verification are specified in Table 5.5.			
Test Object	Indexing	Hypothesis Verification (Matching)	
	♯Hypotheses	♯Accepted	♯Rejected
hatchback	69	0	69
sedan	71	0	71
wagon	60	0	60
sports-car	87	0	87
faceA	86	0	86
faceB	49	0	49
bcirc	4	0	4
jar	66	0	66
mushroom	8	0	8
pipe	40	0	40
pot	20	0	20
taco	11	0	11
top	29	0	29
vase	74	0	74
egg	0	-	-
semi	0	-	-

white contours superimposed on the intensity images) were manually edited to extract

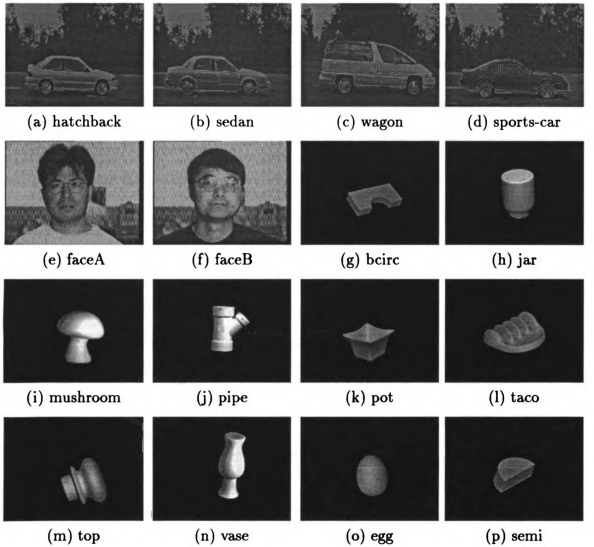


Figure 5.5: Objects for recognition (not existing in the database).

vehicle silhouettes for indexing, but the entire edge contours were used for matching; the indexing scheme produced several model hypotheses on *taurus* and *camaro*, but the matching scheme refuted these hypotheses because of their large fitting errors. For test objects (not having similar models in the database) in Figure 5.5(e)-(n), the recognition system failed to recognize them as indicated in the third column of Ta-

ble 5.10. Two face images were also manually edited to extract face silhouettes for indexing. Note that in the last two rows of Table 5.10, the recognition system did not generate any hypothesis for test objects *egg* and *semi*. Consequently, the recognition system did not need to perform any verification test for these two objects as indicated by the symbol $-$. These two objects are automatically rejected as new objects at the indexing phase. The results in Table 5.10 indicate that the recognition system does not produce any false alarm for these 16 test objects.

Results with occluded objects are given in Table 5.11: the notation is the same as in Table 5.10. The first row of Figure 5.6 shows four test objects *swan13*, *phone1*,

Table 5.11: Recognition results for 4 occluded objects.

Indexing attribute tolerance criteria $(e_\varsigma, e_\tau, e_\rho, e_{\phi_1}) = (4.0, 0.4, 4.0, 0.005)$ Parameters for hierarchical verification are specified in Table 5.5.			
Test Object	Indexing	Hypothesis Verification (Matching)	
	♯Hypotheses	♯Accepted	♯Rejected
swan13	58	0	58
taurus1	62	0	62
phone1	86	0	86
cup	58	0	58

taurus1, and *cup*; the second row of Figure 5.6 gives the extracted parts of these four objects; the third row of Figure 5.6 shows the extracted parts of the corresponding models. As can be seen in the second and third rows of Figure 5.6, the occlusion effect makes the extracted parts completely different, causing the indexing scheme to fail to produce the corresponding hypotheses for verification test. Table 5.11 shows that the recognition system fails to recognize these four objects. The last row of Figure 5.6 depicts the results of model fitting, demonstrating that these four objects

do have corresponding models in the database and that the matching stage can verify this. The recognition system has false dismissal error for these four occluded objects. Additional examples of false dismissal were also given in the first 5 rows of Table 5.8. Thus, the recognition system breaks if the indexing scheme fails to generate correct model hypotheses for verification; this only occurs when all the parts of an observed object are occluded.

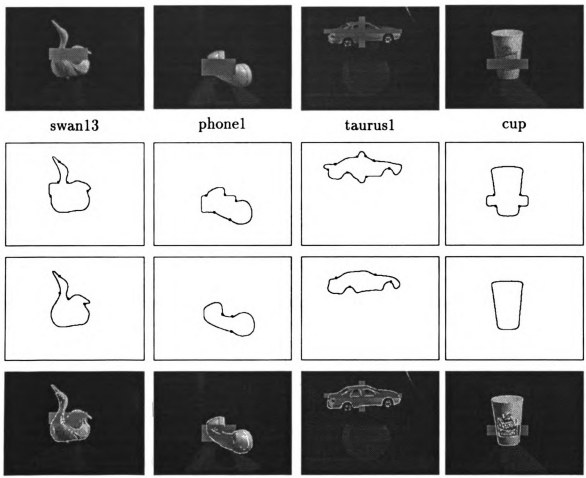


Figure 5.6: Examples of false dismissal. Row 1 contains four test objects. Row 2 presents the parts extracted from the test objects. Row 3 gives the parts extracted from the models. Row 4 shows the results from model fitting.

5.2 Motion Tracking

Motion detection and tracking of moving objects from image sequences are important capacities for any vision system designed to operate in an uncontrolled environment. Tracking in computer vision, however, is still in the development stages and has had few applications in industry. It is hoped that tracking combined with other technologies may produce effective vision systems. Important applications which would be impacted by a successful tracking system include (1) military target acquisition, (2) military and commercial surveillance, (3) traffic monitoring, and (4) robotics. For example, recognizing and tracking parts on a moving conveyor belt would enable robots to pick up the correct parts in an unconstrained environment.

In this work, we consider tracking with a stationary camera. Our assumption is that only the object of interest is in motion and other objects are considered as still background. Section 5.2.1 contains a brief review of some of the previous work related to tracking. A description of the proposed tracking method in given in Section 5.2.2. A motion segmentation scheme for extracting edges of the moving object is presented in Section 5.2.3. In Section 5.2.4, we show the results of tracking objects in three image sequences.

5.2.1 Previous Work

In general, there are two approaches toward the problem of tracking moving objects from image sequences. One is *recognition-based tracking* and the other is *motion-based tracking*. Recognition-based tracking (e.g., [1, 18, 77]) is actually a modification of

object recognition. This approach attempts to recover the pose of an object at each frame. Once correspondences between image and model features are determined, changes in positions of image features in successful frames are used to update the pose of the object. To accurately recovered the object pose at each frame requires an exact *geometric model* of the object. Moreover, the tracked object has to be recognizable. Since object recognition is a high-level operation that can be costly to perform, the performance of the tracking system is limited by the efficiency of the recognition system as well as the types of objects that are recognizable.

Motion-based tracking systems differ significantly from recognition-based systems in that they depend completely on motion detection to track the moving object. They have the capability of tracking any moving object regardless of size or shape. Motion-based techniques (*e.g.*, [2, 84]) can be further subdivided into ***feature-based tracking*** and ***optical flow tracking.*** Feature-based tracking requires stable feature extraction prior to establishing correspondences between the model features and the sensed features in consecutive frames. Correspondence establishment may become difficult if the image includes many objects and feature extraction is ambiguous. On the other hand, gradient-based tracking needs additional constraints (*e.g.*, ***smoothness*** [57]) to compute the optical flow. Although this technique produces good results for smooth object surfaces, the constraints increase the estimation error around the object boundary. Both techniques are sensitive to noise and do not yield satisfactory results for the real images typically encountered in practical applications.

5.2.2 Proposed Method

The proposed approach to the motion tracking problem is recognition-based. The pose estimation is carried out by the Newton's method with Levenberg-Marduardt minimization (N-L-M) to force a convergence. The N-L-M method requires a good initial estimate of parameters that is relatively close to object pose to avoid converging to a local minimum. For tracking an object over a sequence of frames, an automated guess of initial estimate of parameters can be acquired by (1) using the pose estimate from the previous frame plus a velocity estimate for each parameter derived from the previous two frames [77], or (2) by using the pose estimate from the previous frame (if the motion of the object is small and smooth) and renewing the iterations of the N-L-M method. We have selected the latter because the N-L-M method can recover the translation parameters within a few iterations regardless of the initial estimate of translation parameters, and a coarse object view (orientation) is sufficient to form a good initial parameter estimate for the N-L-M method. However, the tracking system needs to pay attention to the situation when the object being tracked moves across aspects. In this case, the model aspect for the previous frame can no longer be used to fit the current frame, but the pose estimate from the previous frame can still be used as a good initial parameter estimate. The problem can be solved by using the mean-squared distance error (MSE) of the fitting as an indicator to determine if a new model aspect should be introduced. For example, suppose that the system initially uses model aspect M_i to track a moving object. The system continues to use M_i for tracking the object until the fitting error at a particular frame t_j is over

the user's specified error threshold E_t. This indicates that the system should change model aspect. The system can use the silhouette of the tracked object at frame t_j to index into the database to generate model-aspect hypotheses, among which the first neighboring model aspect of M_i is chosen to continue the tracking. The system selects the next lower ranked neighboring model aspect if the previous one fails. This procedure continues until all frames in the image sequence are tracked.

In fact, we can use the pose estimate of the previous frame plus a velocity estimate obtained from previous two frames to predict the pose of the current frame; if the pose estimate is out of the aspect boundary of the current model aspect, we can use the motion information obtained to select the neighboring model aspect to continue the tracking. This method definitely outperforms the aforementioned method because the neighboring model aspect can be automatically selected without resorting to indexing. This is future work to improve the tracking system.

5.2.3 Motion Segmentation

For the tracking system to be effective, we need to obtain edgemaps of the tracked object from images. The use of motion information for object segmentation from image sequences has been demonstrated as a successful technique when under the assumption that the images contain only one single moving object. Thus, our implementation is based primarily on this technique.

Edgemaps of the images are created using the Canny edge detector [25]. The background noise edges are excluded by ANDing a mask on the edgemap. Such a

mask is generated through motion segmentation. In practice, the simplest implementation of motion segmentation is image subtraction, in which the image from the previous frame is subtracted, pixel by pixel, from the image of the current frame. When images are subtracted from one another, the absolute intensity difference of the background tends to be very small while the absolute intensity difference along the object boundary is large as compared to that of the background. An appropriate threshold to segment an intensity-subtracted image into a binary image can be obtained by a simple Gaussian thresholding technique [62]. The resulting binary image is dilated several pixels in all directions to ensure that the mask includes the object boundary. The mask may cover small portions of background edges. As a consequence, the obtained edgemap may contain some small background edge contours. Since the N-L-M method can handle partially occluded objects, the generation of the mask need not be perfect.

The motion segmentation technique proposed by Dubuisson and Jain [38] is adopted to generate the mask. The basic idea is as follows. Let f_1, f_2, and f_3 be three image frames extracted from the image sequence at times t_1, t_2, and t_3 respectively. They are blurred by a 3×3 averaging mask to smooth out the background and to preserve the *ramp* edges around the moving object. Let $\hat{f}_1$, $\hat{f}_2$, and $\hat{f}_3$ represent the blurred images of f_1, f_2, and f_3, respectively. The difference image between frames 1, 2, and 3 is then computed as follows:

$$D(i,j) = |\hat{f}_1(i,j) - \hat{f}_2(i,j)| * |\hat{f}_2(i,j) - \hat{f}_3(i,j)|$$

where '$*$' denotes the bitwise multiplication operation and $|\cdot|$ denotes the absolute value operator. This difference image D enhances the pixels around the boundary of the moving objects. Let t_d^{low} and t_d^{high} be two thresholds specified by the user. The difference image D is then thresholded at t_d^{low} to keep only the pixels where a significant intensity change occurs. This thresholding procedure produces an image D_t defined by:

$$D_t(i,j) = \begin{cases} 1 & \text{if } D(i,j) > t_d^{low} \\ 0 & \text{otherwise} \end{cases}$$

Connected components are then extracted from D_t using an 8-connected neighborhood. A connected component is removed if none of its pixels has a value larger then t_d^{high}. This cleaning procedure generates an image C with "on" pixels around the boundary of the moving object. The next step is to dilate the components in C until all isolated components are connected into one single connected component using a chamfering technique to record the distance of dilation. The internal holes of this single connected component are then filled with "on" pixels. The final step is to erode the connected component on the boundary back to its original size. The information for erosion is embedded in the chamfered distance image. The resulting image is a mask covering the moving object.

Figure 5.7 presents the results of the described motion segmentation algorithm using three image frames from an image sequence containing a moving vehicle. Figure 5.7(d) shows the generated mask in which black pixels are the thresholded regions from motion segmentation and gray pixels are the results from dilation and erosion of the thresholded regions. Figure 5.7(e) shows the edgemap of the vehicle obtained by

using the mask to filter out most of the background. Figure 5.7(f) shows the result of alignment indicated by the white contours overlaid on the image. This demonstrates that the extracted edgemap for alignment need not be perfect; however, for indexing, the boundary of the edgemap should be as good as possible to avoid testing many false hypotheses; nevertheless, indexing is not required if the object does not move across aspects or if the motion information is incorporated in the tracking system to select neighboring aspects.

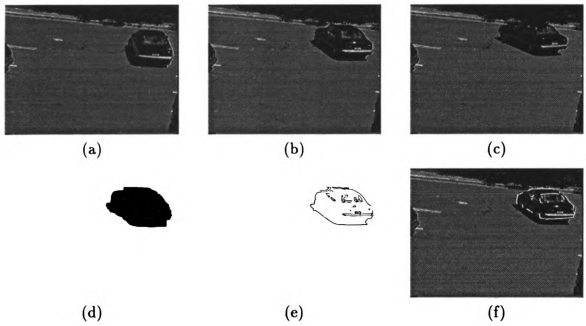


Figure 5.7: Results of the motion segmentation. (a) Frame 1. (b) Frame 2. (c) Frame 3. (d) Mask for frame 2. (e) The edgemap of frame 2 obtained by using the mask to exclude the background. (f) The fitted edgemap (from the N-L-M method) superimposed on frame 2.

5.2.4 Experimental Results

In this section, we intend to demonstrate the viability of the tracking system on both indoor and outdoor moving objects and explain how the tracking system handles the object moving across aspects. The motion tracking method was tested on three image sequences. Two of them were simulated image sequences taken in the lab to study the validity and accuracy of the method. One image sequence taken from a moving vehicle at a parking lot was used to demonstrate the applicability of the method.

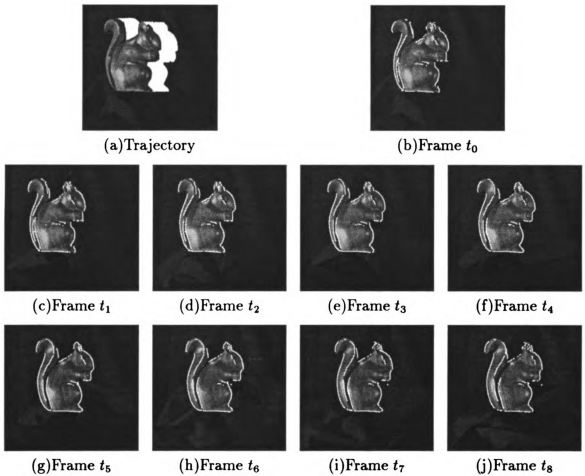
Tracking A Moving Squirrel Carving

The first experiment was conducted with an image sequence from a moving squirrel carving as shown in Figure 5.8. The image sequence was generated using a camera mounted on a translation stage and the object on a rotating base to simulate the motion. On every image frame grabbed, the translation stage was moved $5mm$ along the X axis and the rotating base was rotated 2 to 5 degrees counterclockwise. Table 5.12 shows the results of the tracking. The model aspect and the initial parameter estimates for the first frame were provided by the indexing scheme. The initial parameter estimates of the subsequent frames were obtained from the fit parameter estimates of the previous frame. The final column of Table 5.12 illustrates that tracking was achieved with 1.3 pixels MSE on the edgemaps within 8 iterations of the N-L-M method. From the results shown in Table 5.12, it is not difficult to see that a smaller angle of rotation α results in a smaller MSE but not necessarily small errors in 3D pose parameters, which is consistent with the error analysis presented in Sec-

tion 2.5.1. The aligned model edgemaps are overlaid on the image sequence as shown in Figure 5.8. Figure 5.9 depicts the evolution of the parameter estimates within and across frames. It is obvious that the fit translation parameters are consistent with the trajectory of the motion as indicated in Figure 5.8(a). The parameters of the direction of rotation axis, θ and ϕ, are less sensitive, especially when the rotation angle, α, is small (for example, see the E_θ values at frames t_4, t_5, and t_6 in Table 5.12). This is because small α results in small error (see Section 2.5.1) and small error leads to the insensitivity of the related rotational parameters (See Section 3.3.3). This explains why the errors for the rotation axis, E_θ and E_ϕ, are large for these three frames. For the same reason, these three frames tend to need more iterations for convergence because the N-L-M method may have to wander in the parameter space (due to the random walk) before convergence.

Table 5.12: Results of motion tracking for a squirrel carving.

Frame♯	α	E_{t_x}	E_{t_y}	E_s	E_α	E_θ	E_ϕ	♯Iteration	MSE (pixel)
t_0	$-15°$	1	0	0.0	1.1°	1.0°	0.0°	4	1.22
t_1	$-13°$	1	0	0.0	1.6°	1.0°	0.0°	4	1.20
t_2	$-8°$	1	0	0.0	2.5°	3.0°	0.4°	7	1.18
t_3	$-5°$	1	0	0.0	3.5°	1.7°	0.1°	4	1.09
t_4	$-2°$	0	0	0.0	2.0°	6.8°	2.5°	8	0.98
t_5	4°	1	0	0.0	0.3°	9.2°	1.7°	7	0.99
t_6	7°	0	0	0.0	0.5°	9.4°	6.5°	7	1.19
t_7	12°	1	1	0.0	0.2°	2.2°	4.9°	4	1.21
t_8	15°	1	0	0.0	0.4°	1.5°	3.1°	6	1.27
$(t_x, t_y, s, \alpha, \theta, \phi)$ represents the pose parameters. $E_p = \lvert\hat{p} - p_{true}\rvert$ where $\hat{p}$ is the estimated value of the ground truth p_{true}. The measurement unit for E_{t_x} and E_{t_y} is pixels.									



(a)Trajectory (b)Frame t_0

(c)Frame t_1 (d)Frame t_2 (e)Frame t_3 (f)Frame t_4

(g)Frame t_5 (h)Frame t_6 (i)Frame t_7 (j)Frame t_8

Figure 5.8: Motion tracking of image sequences of a moving squirrel. (a) Image of frame t_0 added with thresholded images of frame t_1 - t_8. (b)-(j) are the results of model fitting with the white contours indicating the fitted edgemaps.

Tracking Across Aspects

The second experiment was conducted with an image sequence generated from a moving Ford Taurus 1:20 scaled toy model as shown in Figure 5.10. The Taurus model was mounted on the tripod (see Figure 2.7) to simulate the motion across two neighboring aspects diagonally, i.e., from aspect *taurus1* to aspect *taurus20*. Ten test images were generated by equally sampling the orientations between these two neigh-

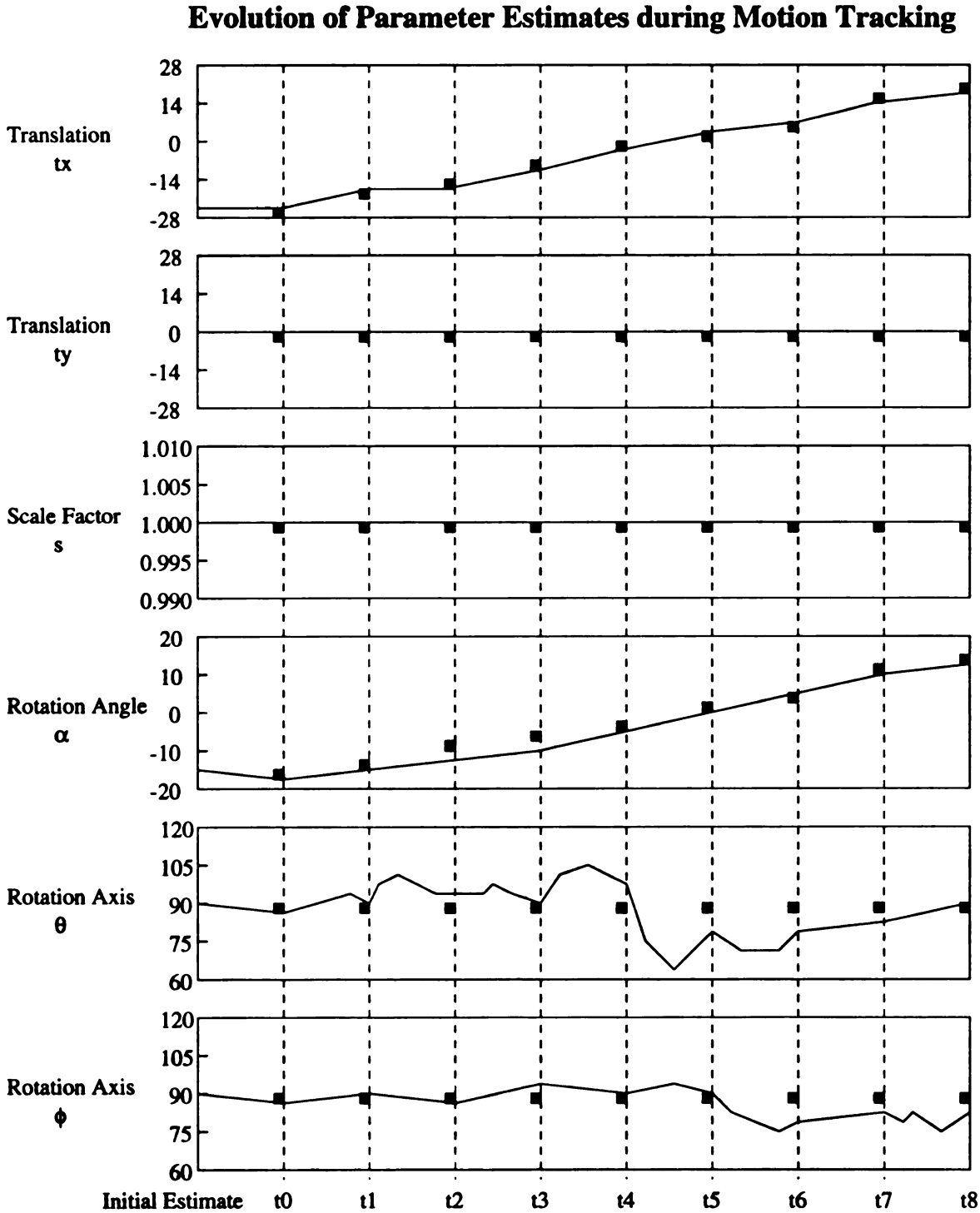


Figure 5.9: The evolution of parameter estimates during motion tracking of the squirrel. The black square boxes indicate the ground truth parameters. The vertical dotted line denote the convergence of the N-L-M method at a particular frame.

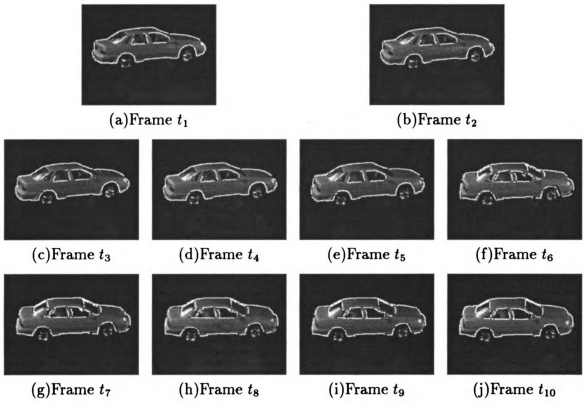


Figure 5.10: Motion tracking on the Taurus image sequence. The white contours indicate the fitted edgemaps overlaid on the image sequence.

boring aspects. In this image sequence, frames t_1–t_5 belong to aspect *taurus1* while frames t_6–t_{10} belong to aspect *taurus20*. The aligned model edgemaps are superimposed on the image sequence as shown in Figure 5.10. The goal of this experiment is to study how the system handles the situation when object motion is across two neighboring aspects of a model. Figure 5.11 shows the results of tracking without changing model aspects. Both curves indicate that the fitting error degrades significantly when the system uses the wrong model aspect for tracking. The intersection of these two curves suggests the equilibrium point where the system should change the model aspect to continue tracking the object. This raises questions about (1)

when such a decision should be made, and (2) which neighboring aspect should be used to continue the tracking. The answers lie in Table 5.13 which lists the first three ranked model-aspect hypotheses and their fitting errors (MSE). The correct model-aspect hypotheses for frames t_1–t_{10} are either ranked as 1 or 2 from indexing. The fitting error is getting larger as the vehicle approaches the aspect boundary which is in between frames t_5 and t_6. Suppose that 2.2 pixels MSE is the user's specified error threshold for the tracking system to change from one model aspect to another. Without any prior knowledge, the first frame (i.e., frame t_1), has to resort to the recognition system to find out the object model aspect and identity. For frame t_1, the indexing scheme generates 19 model-aspect hypotheses, and the verification stage accepts the first model-aspect hypothesis, *taurus1*, as the correct model aspect because the MSE is below the user's specified error threshold. The system uses *taurus*1 to continue tracking the vehicle without resorting to indexing until frame t_6 where the MSE is above the user's specified error threshold (see Figure 5.11); then the system uses the object silhouette at frame t_6 for indexing; since model aspect *taurus20* is the first hypothesized model aspect neighboring to *taurus1*, the system selects it for verification; the 1.8 pixels MSE of fitting *taurus20* to frame t_6 indicates that *taurus20* is the right model aspect, so the system uses it to continue the tracking. The results of this tracking is also depicted in Figure 5.12.

Tracking Using A Real Image Sequence

The image sequence in this experiment consists of an ordered list of frames: $f_1, f_2, \cdots, f_n$. For the purpose of the demonstration, a sequence containing $n = 120$

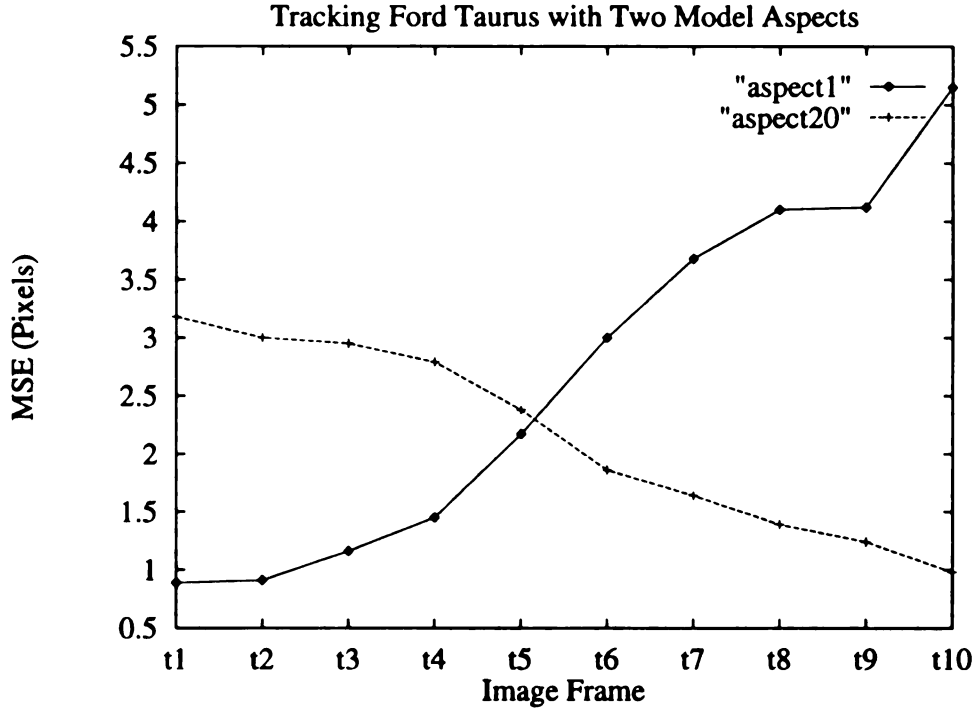


Figure 5.11: Tracking Ford Taurus from two model aspects.

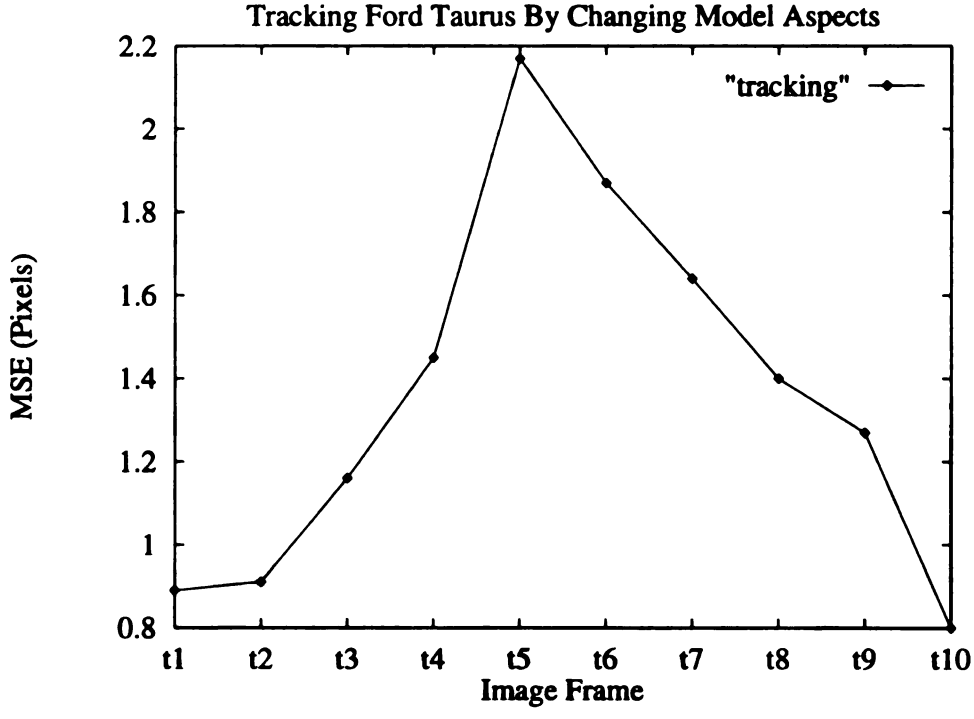


Figure 5.12: Tracking Ford Taurus by changing model aspects.

Table 5.13: Motion tracking across aspects.

N_{HG} denotes the number of hypotheses generated via indexing. Indexing attribute tolerance criteria: $(e_\varsigma, e_\tau, e_\rho, e_{\phi_1}) = (4.0, 0.4, 4.0, 0.005)$ MSE (unit = pixels) denotes the fitting error in the N-L-M method.								
Frame No.	N_{HG}	Rank 1		Rank 2		Rank 3		Correct Model
		Model	MSE	Model	MSE	Model	MSE	
t_1	19	taurus1	**0.9**	taurus10	5.2	taurus9	5.5	taurus1
t_2	21	taurus1	**0.9**	taurus10	5.3	taurus18	4.3	taurus1
t_3	20	taurus1	**1.1**	taurus18	4.9	mug26	10.1	taurus1
t_4	20	taurus1	**1.4**	taurus18	5.1	mug26	10.2	taurus1
t_5	23	taurus1	**2.1**	taurus18	6.1	mug26	10.6	taurus1
t_6	28	phone1	14.1	taurus20	**1.8**	mug16	8.3	taurus20
t_7	27	phone1	15.0	taurus20	**1.6**	mug16	8.6	taurus20
t_8	27	taurus20	**1.4**	mug34	12.8	mug25	12.0	taurus20
t_9	31	taurus20	**1.2**	mug5	11.3	mug15	9.0	taurus20
t_{10}	24	taurus20	**1.0**	zebra23	13.5	camaro11	7.4	taurus20

frames was used as a test (see Figure 5.13). This image sequence was captured by a cam-corder and stored on a video tape. The object is a Ford Taurus vehicle moving around a parking lot. One frame out of ten was used to perform the tracking ($f_{t_1}, f_{t_2}, \cdots, f_{t_{12}}$, which is a total of 12 frames). Figure 5.13 shows these twelve selected frames and the result of the tracking.

Note that although the motion segmentation provides satisfactory masks, it is still difficult to automatically extract the exact boundary of the vehicle, especially the boundary underneath the vehicle caused by the shadow, as can be seen in Figure 5.7(e). The imperfect boundary extraction may render the indexing scheme ineffective. We manually cleaned the top portion of the boundary and leave the bottom portion untouched. The corresponding model aspects for these twelve frames via indexing are given in Table 5.14. The fitting error (MSE) for each frame is also given

in Table 5.14. As mentioned in Chapter 3, in order to allow a full 3D rotation of an object, we need to recover the roll angle of the image plane such that the model can brought into alignment with the observed object. This roll angle as discussed in Chapter 4 can be recovered from the corresponding part features of the model and the sensory data, but we have not implemented this stage yet in the current system. This will be left for the future work. As a result, the fitted edgemaps for these 12 frames are a little tilted as shown in Figure 5.13, and the MSE values as indicated in Table 5.14 are a little large as compared to the MSE values for those simulated ones in the previous two experiments.

Table 5.14: Results of tracking the Ford Taurus.

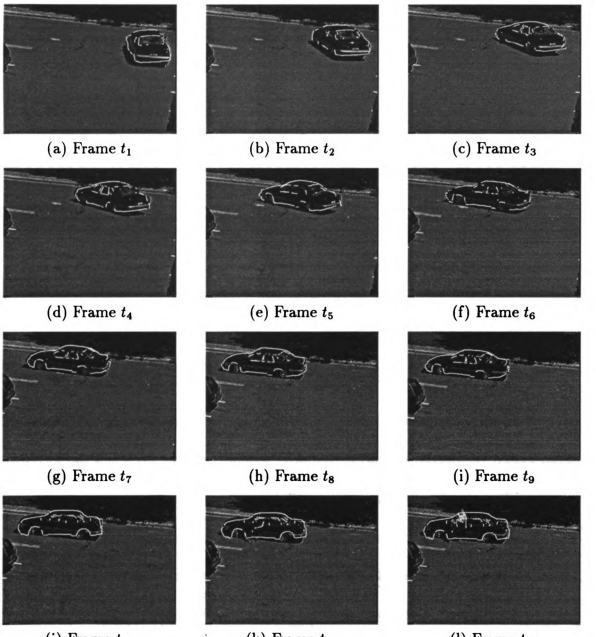
Frame	Model Aspect	MSE (pixels)	Frame	Model Aspect	MSE (pixels)
t_1	taurus15	1.9	t_7	taurus12	2.2
t_2	taurus15	2.2	t_8	taurus12	1.9
t_3	taurus14	2.1	t_9	taurus11	1.7
t_4	taurus14	1.7	t_{10}	taurus11	2.1
t_5	taurus13	1.8	t_{11}	taurus11	2.4
t_6	taurus12	2.5	t_{12}	taurus11	2.3

5.3 Summary

We have combined the modeling, matching, and indexing modules together with a hierarchical verification strategy into an integrated recognition system. The hierarchical verification strategy uses three sets of threshold values in the matching module to quickly remove from further verification the false model-pose hypotheses generated from the indexing module. This hierarchical verification strategy has shown to be vi-

tal and efficient in expediting the recognition, especially when the indexing module fails to reduce the number of verifications required due to the partial occlusion of the observed object. Our recognition experimental results have shown that we have made good progress toward a general alignment system. This system should work well on some existing recognition problems.

We have demonstrated the applicability of the recognition system for tracking single moving objects in image sequences. We have also studied how the tracking system handles the object moving across model aspects. Our experimental results from both simulated and real image sequences have shown the viability of the tracking method. However, for the tracking method to work really well on real image sequences, the current system needs to recover the roll angle of the image plane to allow a full 3D rotation of the moving object. This is left as the future work.



(a) Frame t_1 (b) Frame t_2 (c) Frame t_3

(d) Frame t_4 (e) Frame t_5 (f) Frame t_6

(g) Frame t_7 (h) Frame t_8 (i) Frame t_9

(j) Frame t_{10} (k) Frame t_{11} (l) Frame t_{12}

Figure 5.13: Tracking a moving vehicle in a parking lot.

Chapter 6

Summary, Conclusion, and Future Research

6.1 Summary and Conclusions

The research addressed in this thesis dealt with recognizing arbitrary curved 3D objects from single 2D intensity images. We have developed a complete object recognition system within the alignment paradigm. This paradigm involves three major schemes. The modeling scheme consists of constructing model aspects for predicting the object appearance seen from any viewpoint. The indexing scheme generates hypotheses about candidate model aspects and poses which are then verified in the matching scheme to support/refute the hypotheses and also to estimate/refine the object pose of the correct model.

In Chapter 2, we approached the problem of modeling 3D sculptured objects using multi-view $2\frac{1}{2}$D representations. The modeling scheme adopted the curvature

method of Basri and Ullman [9] to generate the predicted appearance of a 3D object with a smooth surface following a 3D rotation. In this modeling scheme, a 3D object is represented by a number of view-centered model aspects, rather than by a single object-centered geometric model. Each model aspect covers a range of many possible neighboring viewing angles within the aspect. Using a number of model aspects, the contour of an object seen from any viewpoint can be predicted. The computations required in this modeling scheme during the prediction stage is simple, and the model construction is almost automatic when object prototypes are available. This modeling scheme was applied to 20 arbitrary curved 3D objects with 578 model aspects, and the experimental results are supportive of the design. We conclude that this modeling scheme is viable in modeling many kinds of objects including both polyhedra and sculptured objects.

In Chapter 3, we studied the problem of pose estimation from a known model aspect and a 2D edgemap extracted from the scene. The proposed pose/refinement algorithm, which can handle objects with partial occlusion, does not assume the existence of salient features in the image, and thus, is directly applicable to smooth objects with sculptured surfaces. Due to the unavailability of salient features in objects with sculptured surfaces, we developed an iterative matching technique using the Newton's method with Levenberg-Marquardt minimization to estimate/refine the object pose in a hill-climbing manner. As with any minimization technique, this iterative matching technique requires good initial parameter estimates to avoid converging to a local minimum. Two heuristics were adopted to improve the pose accuracy and handle occlusion. The first heuristic was used to synthesize feature correspondences

for pose estimation/refinement of objects with partial occlusion. The second heuristic was adopted to to maintain a balance of alignment between object internal edges and object silhouettes. Many conclusions were drawn from this study. The matching algorithm was applied to 6 test objects, either partially occluded or non-occluded, with 240 samples of initial parameter estimates in each object experiment. The results from these Monte Carlo experiments support the hypothesis that fitting internal edges and silhouette simultaneously can produce more accurate estimation of pose parameters than fitting the silhouette alone. The results also demonstrate a high rate of convergence for a broad set of starting orientations within a model aspect. The matching algorithm was also applied to 60 test model aspects, and the results indicated that alignment was achieved with 1.6 pixels mean-squared distance error on the edgemap within 10 iterations of the hill-climbing procedure. These results are consistent with those reported by Lowe [76]. We conclude that large number of fine aspects are not needed for modeling. The experimental results also demonstrate that arbitrary curved objects can be handled and some occlusion can be tolerated. These results are not tied to the modeling scheme from Chapter 2. The matching algorithm can work with a feature-based CAD model which can predict an object's edgemap when given a viewpoint.

In Chapter 4, we studied the problem of indexing candidate model aspect and pose from a model-aspect database using 2D silhouettes. A part segmentation scheme was proposed to decompose silhouettes into parts whose invariant attributes are used for indexing. A hypothesis grouping scheme was also proposed to cluster hypotheses stemming from the same model. We also studied four voting schemes for ordering

hypotheses based on prior knowledge of pre-stored models and the visual evidence of the observed objects such that the most likely hypotheses are verified first. When combined into a complete system, these techniques make progress toward improving accuracy and efficiency by pruning false model hypotheses and minimizing unnecessary verification tests. Several experiments were performed on a database containing 658 model aspects, and the results support the hypothesis that HYBRID is the most efficient hypothesis ordering scheme in pruning false model hypotheses. Experimental results also indicate that the indexing scheme generates many hypotheses, but, through hypothesis grouping and ordering, a significant number of verification tests for false models can be avoided. The success of the indexing scheme also indicates that the proposed part representation is robust under global transformation and local deformation.

We have combined the modeling, matching, and indexing schemes into a recognition-by-alignment paradigm together with a hierarchical verification strategy. The hierarchical verification strategy uses three sets of threshold values in the matching scheme to quickly eliminate from further consideration the false model hypotheses generated from the indexing scheme. The matching algorithm was applied to 100 randomly selected model aspects with totally 1200 test cases to select appropriate threshold values at various levels. These empirical threshold values were used to perform the recognition experiments on 60 test model aspects. The results indicate that the integrated recognition system is viable in handling a large database of many kinds of objects and the hierarchical verification strategy is effective in expediting the recognition, especially when the indexing scheme fails to reduce the number of false

model hypotheses generated due to partial occlusion of the observed object.

We have applied the recognition system to tracking a single moving object in a scene from an image sequence. We have also studied how to track the moving object across aspects. The recognition system was applied to 3 either simulated or real image sequences to perform object tracking. Experimental results indicated the viability of the tracking method and also demonstrated the applicability of the tracking method by tracking a real car in an image sequence.

In this thesis, we have proposed a model-based computational paradigm for recognition of arbitrary curved 3D objects from single 2D intensity images. Experimental results show that this computational paradigm is viable in handling a large database of many kinds of objects including polyhedra and sculptured objects.

6.2 Future Research

We have presented a model-based object recognition paradigm which shows that recognizing 3D objects with sculptured surfaces from 2D images is feasible. However, there are still a number of research issues to be addressed in the future. The most prominent one is the figure-ground separation. In this thesis, we have implicitly assumed a reasonable extraction of the object from its background. Figure-ground separation is a major obstacle to cognitive psychology and computer vision in general and to the alignment in particular. Thus, future work should include an investigation of a better figure-ground separation algorithm.

In Chapter 2, we have used a 3-image stereo matching algorithm to compute the

curvature and 3D information from edgemaps for object modeling. This 3-image stereo matching technique has applied several constraints/heuristics to eliminate ambiguous matches of object internal edges. However, the predicted appearance of objects shown in the experiment results indicates that there are still some outliers due to inaccurate matches in this stereo matching algorithm. Thus, the constraints/heuristics used in this stereo matching algorithm for object modeling should be further studied.

In Chapter 3, we have proposed a matching algorithm for pose estimation/refinement. We have defined a merit function based on the mean-square distance error between the observed edgemap and the edgemap of a model aspect. We have seen that this matching algorithm requires reasonable initial parameter estimates to avoid converging to local minima. We have incorporated the random walk feature in the matching algorithm to escape local minima. However, it is a stochastic process, which may not always lead to a global minimum convergence. Moreover, this random walk feature sometimes causes the matching algorithm to wander around in parameter space, resulting in a slow convergence. Thus, it might be better to remove this random walk feature in the matching algorithm if a good initial parameter estimate can be obtained for pose refinement or, perhaps to involve some other kind of search through parameter space. A good initial parameter estimate may be obtained through a coarse-to-fine subdivision of the parameter space; the mean-square distance error may be used as a similarity measure to determine the best initial parameter estimate. All these require further investigation.

In Chapter 4, we have proposed an indexing scheme for generating model hy-

potheses from a database. This indexing scheme uses shape features obtained from the object silhouette for indexing and is effective in pruning false model hypotheses; however, the discriminating capacity of shape features is not very powerful. It is likely to be useful to incorporate other types, such as color or texture features, or shape features and relational features from internal edge contours. Also, it might be a good idea to design a similarity measure between shape features for a better hypothesis ordering scheme to avoid unnecessary verification tests.

The use of part invariant features derived from object silhouettes enables the indexing scheme to effectively generate a small set of candidate model hypotheses; but this indexing scheme fails when all parts of an observed object are partially occluded. This suggests that the silhouette may need to be segmented using a small scale for smoothing; however, this could also lead to more noisy and unnecessary parts which may degrade the indexing. Toward this end, a scale-space approach to part segmentation might be an important issue to be explored in the future.

The proposed object recognition paradigm in this thesis assumes that an object's orientation is fixed when the object is modeled on a viewsphere. Consequently, the matching technique assumes a zero-roll angle of the image plane. This assumption fixes one degree of freedom of the rotation and has no doubt and contributes to the accuracy of the pose reported. The detection of this roll angle can be done by least-square solution using part correspondences between the model and the image in the indexing scheme. Thus, it is desirable to incorporate this roll angle detection in the indexing scheme and then to use the detected roll angle for the matching algorithm to allow a full 3D rotation of an object.

Finally, we have applied the integrated recognition system to track a single moving object in an image sequence. The tracking method applies the indexing scheme to select the model aspect for tracking when the observed object moves across aspects. Although the indexing scheme can effectively generate the candidate neighboring model-aspect hypotheses, this approach is clumsy. Thus, future work should incorporate the motion information detected during the tracking to determine the neighboring aspect automatically without resorting to indexing.

Bibliography

[1] Y. Aloimonos and D. Tsakiris. On the mathematics of visual tracking. *Image Vision Computing*, 9(4):235–251, 1991.

[2] Y. Aloimonos, I. Weiss, and A. Bandopadhay. Active vision. *Int. Journal of Computer Vision*, 2:333–356, 1988.

[3] H. Asada and M. Brady. The curvature primal sketch. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 5(1):2–14, 1983.

[4] N. Ayache and O. D. Faugeras. HYPER: A new approach for the recognition and positioning of two-dimensional objects. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(1):44–54, 1986.

[5] R. Bajcsy and F. Solina. Three-dimensional object representation revisited. In *Proceedings 1st Int. Conf. on Computer Vision*, pages 231–240, London, 1987.

[6] D. H. Ballard and C. H. Brown. *Computer Vision.* Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[7] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf. Parametric correspondence and chamfer matching. In *Proceedings Int. Joint Conf. on Artificial Intelligence*, pages 22–25, Massachusetts, August 1977.

[8] R. Basri. The alignment of objects with smooth surfaces: Error analysis of the curvature method. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 341–346, Illinois, June 1992.

[9] R. Basri and S. Ullman. The alignment of objects with smooth surfaces. In *Proceedings 2nd Int. Conf. on Computer Vision*, pages 482–488, Florida, December 1988.

[10] J. S. Beis and D. G. Lowe. Learning indexing functions for 3-D model-based object recognition. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 275–280, Seattle, WA, 1994.

[11] P. Besl and N. McKay. A method for registration of 3-D shape. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[12] P. J. Besl and R. C. Jain. Three-dimensional object recognition. *ACM Computing Surveys*, 17(1):75–154, 1985.

[13] I. Biederman. Recognition-by-Components: A theory of human image understanding. *Psychological Review*, 94(2):115–147, 1987.

[14] T. O. Binford. Visual perception by computer. In *IEEE Conference on Systems and Control*, Miami, 1971.

[15] G. Blais and M. D. Levine. Registering multiview range data to create 3D computer objects. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(8):820–824, 1995.

[16] R. C. Bolles and R. A. Cain. Recognizing and locating partially visible objects, the local-feature-focus method. *Int. Journal of Robotics Research*, 1(3):57–82, 1982.

[17] R. C. Bolles and P. Horaud. 3DPO: A three-dimensional part orientation system. *Int. Journal of Robotics Research*, 5(3):3–26, 1986.

[18] A. J. Bray. Tracking objects using image disparity. *Image Vision Computing*, 8(1):4–9, 1990.

[19] T. M. Breuel. Indexing for visual recognition from a large model base. AI Lab Memo, MIT, 1990.

[20] T. M. Breuel. Fast recognition using adaptive subdivisions of transformation space. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 445–451, 1992.

[21] R. A. Brooks. Symbolic reasoning among 3D models and 2D images. *Artificial Intelligence*, 17:285–349, 1981.

[22] J. B. Burns, R. S. Weiss, and E. M. Riseman. View variation of point-set and line-segment features. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(1):51–68, 1993.

[23] A. Califano and R. Mohan. Multidimensional indexing for recognizing visual shapes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(4):373–392, 1994.

[24] O. I. Camps. *PREMIO: The Use of Prediction in a CAD-Model-Based Vision System.* PhD thesis, Department of Electrical Engineering, University of Washington, 1992.

[25] J. Canny. A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[26] C.-C. Chen. Improved moment invariants for shape discrimination. *Pattern Recognition*, 26(5):683–686, 1993.

[27] J.-L. Chen, G. C. Stockman, and K. Rao. Recovering and tracking pose of curved 3D objects from 2D images. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 233–239, New York, June 1993.

[28] Sei-Wang Chen. *3-D Representation and Recognition Using Object Wings.* PhD thesis, Michigan State University, 1989.

[29] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. *International Journal of Image and Vision Computing*, 10(3):145–155, April 1992.

[30] Y. Chen and G. Medioni. Fitting a surface to 3-D points using an inflating balloon model. In *Proceedings IEEE 2nd CAD-Based Vision Workshop*, pages 266–273, Champion, PA, 1994.

[31] R. T. Chin and C. R. Dyer. Model-based recognition in robot vision. *ACM Computing Surveys*, 18(1):67–108, March 1986.

[32] R. Cipolla and A. Blake. The dynamic analysis of apparent contours. In *Proceedings 3rd Int. Conf. on Computer Vision*, pages 616–623, 1990.

[33] D. T. Clemens and D. W. Jacobs. Space and time bounds on indexing 3-D models from 2-D images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(10):1007–1017, 1991.

[34] L. D. Cohen and I. Cohen. Finite-element methods for active contour models and balloons for 2D and 3D images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(11):1131–1147, 1993.

[35] M. Dhome, M. Richetin, J.-T. Lapreste, and G. Rives. Determinination of the attitude of 3D objects from a single perspective view. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(12):1265–1278, 1989.

[36] C. Dorai and A. K. Jain. COSMOS – a representation scheme for free-form surfaces. In *Proceedings of the International Conference on Computer Vision*, pages 1024–1029, Boston, MA, June 1995.

[37] C. Dorai, J. Weng, and A. K. Jain. Optimal registration of multiple range views. In *Proceedings of the 12th International Conference on Pattern Recognition*, volume 1, pages 569–571, Jerusalem, Israel, October 1994.

[38] M.-P. Dubuisson and A. K. Jain. Object contour extraction using color and motion. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 471–476, New York, June 1993.

[39] D. Eggert and K. Bowyer. Computing the perspective projection aspect graph of solids of revolution. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(2):109–128, 1993.

[40] O. Faugeras and M. Hebert. The representation, recognition, and locating of 3D objects. *Int. Journal of Robotics Research*, 5(3):27–52, 1986.

[41] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with application to image analysis and automated cartography. *Communications ACM*, 24(6):381–395, 1981.

[42] P. J. Flynn. *CAD-Based Computer Vision: Modeling and Recognition Strategies.* PhD thesis, Department of Computer Science, Michigan State University, 1990.

[43] P. J. Flynn and A. K. Jain. Bonsai: 3D object recognition using constrained search. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(10):1066–1075, 1991.

[44] D. Forsyth, J. L. Mundy, A. Zisserman, C. Coelho, A. Heller, and C. Rothwell. Invariant descriptors for 3-D object recognition and pose. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(10):971–991, 1991.

[45] H. Freeman. Shape description via the use of critical points. *Pattern Recognition*, 10:159–166, 1978.

[46] P. Giblin and R. Weiss. Reconstruction of surfaces from profiles. In *Proceedings 1st Int. Conf. on Computer Vision*, pages 136–144, London, 1987.

[47] W. E. L. Grimson. *Object Recognition by Computer: The Role of Geometric Constraints.* MIT Press, Cambridge MA, 1990.

[48] W. E. L. Grimson and T. Lozano-Perez. Model-based recognition and localization from sparse range or tactile data. *Int. Journal of Robotics Research*, 3(3):3–35, 1984.

[49] W. E. L. Grimson and T. Lozano-Perez. Localizing overlapping parts by searching the interpretation tree. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 9(4):469–482, 1987.

[50] A. D. Gross and T. E. Boult. Error of fit measurements for recovering parametric solids. In *Proceedings 2nd Int. Conf. on Computer Vision*, pages 690–694, Tampa, FL, 1988.

[51] R. M. Haralick, H. Joo, C.-N. Lee, X. Zhuang, V.G. Vaidya, and M. B. Kim. Pose estimation from corresponding point data. *IEEE Trans. on Systems, Man, and Cybernetics*, 19(6):1426–1446, 1989.

[52] K. Higuchi, H. Delingette, M. Hebert, and K. Ikeuchi. Merging multiple views using a spherical representation. In *Proceedings IEEE 2nd CAD-Based Vision Workshop*, pages 124–131, Champion, PA, 1994.

[53] D. D. Hoffman and W. A. Richards. Codon constraints on closed 2d shapes. *Computer Vision, Graphics, and Image Processing*, 31:265–281, 1985.

[54] D. D. Hoffman and W. A. Richards. Parts of recognition. *Cognition*, 18:65–96, 1985.

[55] R. Horaud. New methods for matching 3D objects with single perspective views. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 9(3):401–412, 1987.

[56] B. K. P. Horn. *Robot Vision.* MIT Press, Cambridge, MA, 1986.

[57] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[58] M. K. Hu. Visual pattern recognition by moment invariants. *IEEE Trans. Information Theory*, 12:179–187, 1962.

[59] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(9):850–863, 1993.

[60] D. P. Huttenlocher and S. Ullman. Object recognition using alignment. In *Proceedings 1st Int. Conf. on Computer Vision*, pages 102–111, London, 1987.

[61] D. W. Jacobs. Space efficient 3D models indexing. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 439–444, 1992.

[62] A. K. Jain and M. P. Dubuisson. Segmentation of x-ray and c-scan images of fiber reinforced composite materials. *Pattern Recognition*, 25(3):257–270, 1992.

[63] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Int. Journal of Computer Vision*, 2(1):322–331, 1987.

[64] D. Keren, D. Cooper, and J. Subrahmonia. Describing complicated objects by implicit polynomials. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(1):38–53, 1991.

[65] T. F. Knoll and R. C. Jain. Recognizing partially visible objects using feature indexing hypotheses. *IEEE Journal of Robotics and Automation*, 2(1):3–13, 1986.

[66] J. J. Koenderink and A. J. van Doorn. Internal representation of solid shape with respect to vision. *Biological Cybernetics*, 32(4):211–216, 1979.

[67] V. Koivunen and J.-M. Vezien. Multiple representation approach to geometric model construction from range data. In *Proceedings IEEE 2nd CAD-Based Vision Workshop*, pages 132–139, Champion, PA, 1994.

[68] M. R. Korn and C. R. Dyer. 3-D multiview object representations for model-based object recognition. *Pattern Recognition*, 20(1):91–104, 1987.

[69] D. J. Kriegman and J. Ponce. On recognizing and positioning curved 3D objects from image contours. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12:1127–1137, 1990.

[70] K. N. Kutulakos, W. B. Seales, and C. R. Dyer. Building global object models by purposive viewpoint control. In *Proceedings IEEE 2nd CAD-Based Vision Workshop*, pages 169–182, Champion, PA, 1994.

[71] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson. Object recognition by affine invariant matching. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 335–344, 1988.

[72] Y. Lamdan and H. J. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *Proceedings 2nd Int. Conf. on Computer Vision*, pages 238–249, 1988.

[73] C.-W. Liao and G. Medioni. Surface approximation of a cloud of 3D points. In *Proceedings IEEE 2nd CAD-Based Vision Workshop*, pages 274–281, Champion, PA, 1994.

[74] C. C. Lin and R. Chellappa. Classification of partial 2-D shapes using fourier descriptors. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 9(5):686–673, 1987.

[75] D. G. Lowe. *Perceptual Organization and Visual Recognition.* Kluwer Academic Publishers, Boston, 1985.

[76] D. G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31(3):355–395, 1987.

[77] D. G. Lowe. Fitting parameterized 3-D models to images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(5):441–450, 1991.

[78] D. Marr. *Vision.* Freeman, New York, NY, 1982.

[79] D. Marr and H. K. Nishihara. Representation and recognition of the spatial organization of three-dimensional structure. In *Proceedings Royal Society of London*, volume 200, pages 269–294, 1978.

[80] T. McInerney and D. Terzopoulos. A finite element model for 3D shape reconstruction and nonrigid motion tracking. In *Proceedings of the International Conference on Computer Vision*, pages 518–523, Berlin, Germany, May 1993.

[81] R. Mohan, D. Weinshall, and R. R. Sarukkai. 3D object recognition by indexing structural invariants from multiple views. In *Proceedings of the International Conference on Computer Vision*, pages 264–268, Berlin, Germany, May 1993.

[82] F. Mokhtarian and A. K. Mackworth. A theory of multi-scale, curvature-based shape representation for planar curves. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14:789–805, 1992.

[83] E. E. Molios. Shape matching using curvature processes. *Computer Vision, Graphics, and Image Processing*, 43:203–226, 1989.

[84] D. Murray and A. Basu. Motion tracking with an active camera. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(5):518–522, 1995.

[85] D. W. Murray and D. B. Cook. Using the orientation of fragmentary 3D edges segments for polyhedral object recognition. *Int. Journal of Computer Vision*, 2(2):153–169, 1988.

[86] C. Nastar and N. Ayache. Fast segmentation, tracking and analysis of deformable objects. In *Proceedings of the International Conference on Computer Vision*, Berlin, Germany, May 1993.

[87] C. F. Olson. Probabilistic indexing for object recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(5):518–522, 1995.

[88] A. P. Pentland. *From Pixels to Predicates.* Ablex, 1986.

[89] A. P. Pentland. Recognition by parts. In *Proceedings 1st Int. Conf. on Computer Vision*, pages 612–620, London, 1987.

[90] T. Poggio and S. Edelman. A network that learns to recognize three-dimensional objects. *Nature*, 343:263–266, 1990.

[91] J. Ponce, D. J. Kriegman, S. Petitjean, S. Sullivan, G. Taubin, and B. Vijayakumar. Representations and Algorithms for 3D Curved Objects Recognition. In *A. K. Jain and P. J. Flynn, editors, Three-Dimensional Object Recognition Systems*, pages 327–352. Elsevier, 1993.

[92] N. S. Raja. *Obtaining Generic Parts from Range Images using a Multi-View Representation.* PhD thesis, Michigan State University, 1992.

[93] A. P. Reeves, R. J. Prokop, S. E. Andrews, and F. P. Kuhl. Three-dimensional shape analysis using moments and Fourier descriptors. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 10(6):937–943, 1988.

[94] P. L. Rosin. Multiscale representation and matching of curves using codons. *Computer Vision, Graphics, and Image Processing*, 55(4):286–310, 1993.

[95] H. Samet. *The Design and Analysis of Spatial Data Structures.* Addison-Welsey, New York, 1990.

[96] J. T. Schwartz and M. Sharir. Identification of partially obscured objects in two and three dimensions by matching noisy characteristics curves. *Int. Journal of Robotics Research*, 6(2):29–44, 1987.

[97] W. B. Seales and O. D. Faugeras. Building three-dimensional CAD/CAM models from image sequences. In *Proceedings IEEE 2nd CAD-Based Vision Workshop*, pages 116–123, Champion, PA, 1994.

[98] R. Sekuler and R. Blake. *Perception.* Alfred A. Knopf, Inc., New York, 1985.

[99] K. Siddiqi and B. B. Kimia. Parts of visual form: Computational aspects. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(3):239–251, 1995.

[100] M. Soucy and D. Laurendeau. A general surface approach to the integration of a set of range views. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(4):344–358, 1995.

[101] F. Stein and G. Medioni. Structural indexing: Efficient 2-D object recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(12):1198–1204, 1992.

[102] F. Stein and G. Medioni. Structural indexing: Efficient 3-D object recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(2):125–145, 1992.

[103] G. C. Stockman. Object recognition and localization via pose clustering. *Computer Vision, Graphics, and Image Processing*, 40:361–387, 1987.

[104] G. C. Stockman. Object Recognition. In *Ramesh C. Jain and Anil K. Jain, editors, Analysis and Interpretation of Range Images*, pages 225–253. Springer-Verlag, 1990.

[105] G. C. Stockman, S. Kopstein, and S. Benett. Matching images to models for registration and object detection via clustering. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 3(3):229–241, 1982.

[106] P. Suetens, P. Fua, and A. J. Hanson. Computational strategies for object recognition. *ACM Computing Surveys*, 24(1):5–61, March 1992.

[107] S. Sullivan, L. Sandford, and J. Ponce. On using geometric distance fits to estimate 3D object shape, pose, and deformation from range, CT, and video images. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 110–115, New York, June 1993.

[108] G. Taubin. Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations, with applications to edge and range image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(11):1115–1138, 1991.

[109] D. W. Thompson and J. L. Mundy. Three-dimensional model matching from an unconstrained viewpoint. In *International Conference on Robotics Automation*, pages 208–220, 1987.

[110] L. W. Tucker, C. R. Feynman, and D. M. Fritzsche. Object recognition using the connection machine. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, 1988.

[111] S. Ullman. An approach to object recognition: Aligning pictorial descriptions. AI Memo No. 931, M.I.T. AI Laboratory, 1986.

[112] S. Ullman and R. Basri. Recognition by linear combinations of models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(10):992–1006, 1991.

[113] R. Vaillant and O. Faugeras. Using extremal boundaries for 3-D object modeling. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(2):157–173, 1992.

[114] T. P. Wallace, O. R. Mitchell, and K. Fukunaga. Three-dimensional shape analysis using local shape descriptors. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 3:310–323, 1981.

[115] A. S. Wallack and J. F. Canny. Efficient indexing techniques for model based sensing. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 259–266, Seattle, WA, 1994.

[116] Y. F. Wang, M. J. Magee, and J. K. Aggarwal. Matching three-dimensional objects using silhouettes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 6(4):513–518, 1984.

[117] M. D. Wheeler and K. Ikeuchi. Sensor modeling, probabilistic hypothesis generation, and robust localization for object recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(3):252–265, 1995.

[118] J. M. Wolfe. *The Mind's Eye.* Freeman and Company, New York, 1986.

[119] D. M. Wuescher and K. L. Boyer. Robust contour decomposition using a constant curvature criterion. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(1):41–51, 1991.

[120] J. H. Yi and D. Chelberg. Rapid object recognition from a large model database. In *Proceedings IEEE 2nd CAD-Based Vision Workshop*, pages 28–35, Champion, PA, 1994.

[121] J. Y. Zheng. Acquiring 3-D models from sequences of contours. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(2):163–178, 1994.